ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Αυτομοτοποίηση θαλάμου με όρνιθες»

Του φοιτητή
Ιωάννη Μπανακάκη
Αρ. Μητρώου: 511069

Επιβλέπων
Ιορδάνης Κιοσκερίδης
Καθηγητής

Ημερομηνία ΜΑΙΟΣ 2025

Τίτλος Δ.Ε. Αυτοματοποίηση θαλάμου με όρνιθες
Κωδικός Δ.Ε. **13249**
Ονοματεπώνυμο φοιτητή Ιωάννης Μπανακακης
Ονοματεπώνυμο εισηγητή Κιοσκεριδης Ιορδάνης
Ημερομηνία ανάληψης Δ.Ε. 30-09-2024
Ημερομηνία περάτωσης Δ.Ε. 31-05-2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

«Στην σύντροφο μου…»

Πρόλογος

Η ιδέα της πτυχιακής εργασίας προέκυψε από την αγάπη που έχω για τα ζώα και την φύση καθώς και την ανάγκη διευκόλυνσης στην κατοχή αυτών. Έχοντας κατά καιρούς στο κτήμα μου κότες, κουνέλια μπαξέ κτλ. και έχοντας αρκετή εμπειρία πάνω σε αυτά κατάλαβα ότι για να ευδοκιμούν αλλά και για να μπορώ να τα απολαμβάνω και να τα φροντίζω όπως πρέπει λόγο του λιγοστού χρόνου ήταν αναγκαία η δημιουργία ενός αυτοματοποιημένου συστήματος το οποίο θα με διευκόλυνε στην φροντίδα που χρειάζονται οι κότες μου στην συγκεκριμένη περίπτωση αλλά και στην ασφάλεια τους σε περιπτώσεις που λείπω ή που υπάρχουν δυσμενείς συνθήκες. Έτσι λοιπόν δημιουργήθηκε αυτός ο Αυτοματοποιημένος θάλαμος με όρνιθες που πέρα από την ευκολία διαχειρίσεις , προστασίας, αυτονομίας και εκσυγχρονισμού του επιπέδου ζωής των πτηνών είναι και άμεσα διαθέσιμος για προσαρμογή σε οποιαδήποτε μικρής η μεγάλης κλίμακας φάρμα ή εγκαταστάσεις κάνοντας πολύ μικρές παρεμβάσεις με άμεση ενημέρωση της κατάστασης του θαλάμου από οποιοδήποτε μέρος του πλανήτη και κατά περιπτώσεις ακόμα και την παρέμβαση απομακρυσμένα. Όλο αυτό φυσικά πρέπει να παρέχει μεγάλη αξιοπιστία και αντοχή στο χρόνο προκειμένου να μην υπάρξουν δυσάρεστες καταστάσεις και για αυτόν τον λόγο πέρα από την κατασκευή του δικού μου προσωπικού θαλάμου που βρίσκεται καθημερινά υπό δοκιμή έγιναν και αρκετές δοκιμές και τεστ σε εργαστηριακό περιβάλλον.

Περίληψη

Στα πλαίσια αυτής της εργασίας λοιπόν, κατασκευάστηκε εξ ολοκλήρου ένα σύστημα “Smart” Αυτοματοποιημένου θαλάμου με όρνιθες με την χρήση ενός esp32c6 microcontroller ως εγκέφαλου του συστήματος Ο οποίος πέρα από την επεξεργαστική του ισχύ Παρέχει και πρόσβαση στο Διαδίκτυο μέσω σύνδεσης wifi δίνοντας έτσι τη δυνατότητα. απομακρυσμένου ελέγχου. Με χρήση διαφόρων αισθητήρων Όπως φωτός ,Υπέρυθρων, Θερμοκρασίας Μπορούμε και ελέγχουμε το περιβάλλον που βρίσκεται ο θάλαμος Καθώς και την κατάσταση του εκείνη την στιγμή. Με διακόπτες Μπορούμε να προγραμματίσουμε Τον Μικροελεγκτή για διαδικασίες, όπως Πότε θα ανοίξει, Πότε θα κλείσει, ή Αν θα είναι αυτόματη η πόρτα. Στην περίπτωση που η πόρτα είναι στην αυτόματη λειτουργία θα ανοίγει η κλείνει λαμβάνοντας δεδομένα από τον αισθητήρα φωτός και έπειτα από κάποιο χρονικό όριο εκείνη θα ανοίγει ή κλείνει. Επίσης με τους διακόπτες αυτούς μπορούμε να προγραμματίσουμε Πότε θα ταΐσει τις κότες, Αν θα ενεργοποιηθεί η λειτουργία προστασίας από καιρικές συνθήκες οπού σε περίπτωση παγετού ή βροχής η πόρτα μένει κλειστή προκειμένου να προστατεύσει τις κότες, πόσες κότες έχουμε Καθώς επίσης και τη σύνδεση με το δίκτυο wifi. Οι λειτουργίες ανοίγματος/κλεισίματος της πόρτας και του ταΐσματος γίνονται και χειροκίνητα με συνεχόμενο πάτημα του ανάλογου διακόπτη .Μια οθόνη, LCD 16 x 2 Μας δείχνει τα όλα τα απαραίτητα δεδομένα σε πραγματικό χρόνο .Καθώς επίσης έχουμε σε αυτά πρόσβαση μέσω Web application εφόσον υπάρχει σύνδεση στο διαδίκτυο . Με δυο αναγνώστες RFID έχουμε εικόνα για πόσες κότες βρίσκονται μέσα η έξω από τον θάλαμο με ειδικό βραχιόλι που φοράνε στο πόδι τους. Υπάρχει επίσης σύστημα ασφάλειας με λαμπτήρα led που να ανάβει σε περίπτωση που τελειώνει το φαγητό , το νερό ή κάποια κότα δεν είναι μέσα στο κοτέτσι όπως και παρόμοιο μήνυμα στην εφαρμογή ειδοποιώντας μας έτσι ότι κάτι δεν πάει καλά. Το σύστημα είναι 100% αυτόνομο με την χρήση μπαταρίας και φωτοβολταικού συστήματος κάνοντας το ιδανικό σε περιπτώσεις οπού δεν υπάρχει τροφοδοσία

«Automatic Chicken Coop»

«Ioannis Banakakis»

Abstract

In the context of this project, a “Smart” Automated Chicken House System was completely built using an esp32c6 microcontroller as the brain of the system, which in addition to its processing power also provides access to the Internet via a wifi connection, thus enabling remote control. Using various sensors such as light, infrared, temperature, we can control the environment in which the coop is located, as well as its current state. With switches, we can program the microcontroller for processes such as when to open, when to close, or whether the door will be automatic. In the case where the door is in automatic mode, it will open or close by receiving data from the light sensor and after a certain time limit, it will open or close. Also with these switches we can program When to feed the hens, If the weather protection function will be activated where in case of frost or rain the door remains closed in order to protect the hens, how many hens we have As well as the connection to the wifi network. The opening/closing functions of the door and feeding are also done manually by continuously pressing the corresponding switch. A screen, LCD 16 x 2 Shows us all the necessary data in real time. As we also have access to them via Web application if there is an internet connection. With two RFID readers we have an image of how many hens are inside or outside the chamber with a special bracelet that they wear on their leg. There is also a security system with an LED lamp that turns on in case the food, water runs out or a hen is not in the coop as well as a similar message in the application, thus notifying us that something is wrong. The system is 100% autonomous using a battery and photovoltaic system, making it ideal in cases where there is no power supply.

Ευχαριστίες

Σε μια πολύ δύσκολη χρονιά για εμένα, όπου έπρεπε να μετακομίσω, άλλαξα δουλειά, είχα ατύχημα και έμεινα από δουλειά και γενικά αντιμετώπισα πολλές πολλές δυσκολίες, Θα ήθελα να ευχαριστήσω τους ανθρώπους που με βοήθησαν να συνεχίσω και να σταθώ στα πόδια μου που δεν είναι άλλοι από την γυναίκα μου που με αντέχει τόσο καιρό, την οικογένεια της που μου έχει σταθεί περισσότερο και από τη δική μου, τον αδερφό μου τον Δημήτρη που μπορεί να μην είναι πραγματικά αδερφός μου αλλά στέκεται δίπλα μου και είναι πάντα εκεί και όλους τους φίλους μου καθώς και την οικογένειά μου που με στηρίζουν σε ότι και να κάνω.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
I2C	Inter-Integrated Circuit
LCD	Liquid Crystal Display
RTC	Real-Time Clock
LDR	Light Dependent Resistor
LED	Light Emitting Diode
RFID	Radio Frequency Identification
DEV	Device
NTC	Negative Temperature Coefficient
IOT	Internet of Things
OTA	Over-The-Air
GPIO	General Purpose Input/Output
BOM	Bill of Materials
WIFI	Wireless Fidelity
RAM	Random Access Memory
ROM	Read-Only Memory
MCU	Microcontroller Unit
DIY	Do It Yourself
TX	Transmit
RX	Receive
SDA	Serial Data
SCL	Serial Clock
GND	Ground
VCC	Voltage Common Collector
ASA	Acrylonitrile Styrene Acrylate
IR	Infrared
UI	User Interface

Κεφάλαιο 1ο: Εισαγωγή

1.1 Αντικείμενο και σκοπός της εργασίας

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι ο σχεδιασμός και η κατασκευή ενός αυτοματοποιημένου θαλάμου με όρνιθες, με στόχο την ευκολότερη και αποδοτικότερη φροντίδα των πτηνών, την ενίσχυση της ασφάλειάς τους και την προώθηση της βιώσιμης και τεχνολογικά ενισχυμένης εκτροφής σε μικρές ή μεγάλες εγκαταστάσεις. Η ιδέα για το συγκεκριμένο έργο προέκυψε μέσα από την προσωπική μου εμπειρία , καθώς η ενασχόληση με ζώα και η καθημερινή φροντίδα τους ανέδειξαν την ανάγκη για μια λύση που θα εξοικονομεί χρόνο και θα προσφέρει ασφάλεια και άνεση στα ζώα, ακόμα και όταν ο ιδιοκτήτης απουσιάζει ή δεν έχει άμεση δυνατότητα παρέμβασης.

Εικόνα 1: chickens With Wifi On Smart Farm Πηγή:smart chicken farm - Αναζήτηση Google

Σκοπός της εργασίας είναι η δημιουργία ενός “έξυπνου” συστήματος, το οποίο με τη χρήση σύγχρονων τεχνολογιών – όπως μικροελεγκτές, αισθητήρες, ασύρματη σύνδεση και απομακρυσμένη διαχείριση μέσω web εφαρμογής – θα προσφέρει αυτοματοποιημένες λειτουργίες όπως άνοιγμα/κλείσιμο της πόρτας, αυτόματο τάισμα, παρακολούθηση καιρικών συνθηκών και κατάστασης των ζώων. Επιπλέον, επιδιώκεται η αξιοπιστία και η ενεργειακή αυτονομία του συστήματος μέσω φωτοβολταϊκής υποστήριξης, καθιστώντας το ιδανικό για εγκαταστάσεις χωρίς σταθερή παροχή ρεύματος.

1.2 Σημασία και Λειτουργία Αυτοματοποιημένου Θαλάμου με Όρνιθες

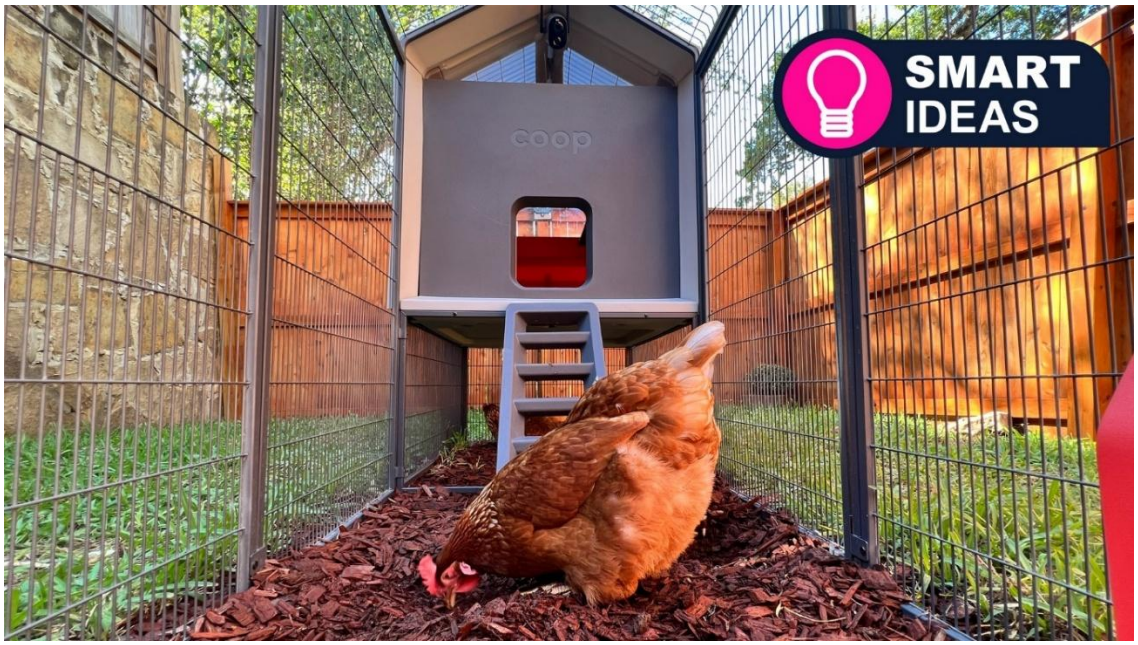
Η ύπαρξη ενός αυτοματοποιημένου θαλάμου για όρνιθες προσφέρει πολλαπλά οφέλη, τόσο για τον ιδιοκτήτη όσο και για τα ίδια τα ζώα. Πρωταρχικά, διασφαλίζεται η σωστή και σταθερή φροντίδα των πτηνών, ανεξαρτήτως της φυσικής παρουσίας του ανθρώπου, μειώνοντας σημαντικά την καθημερινή ενασχόληση και προσφέροντας αυξημένη άνεση και ασφάλεια. Το σύστημα λειτουργεί με έναν μικροελεγκτή ESP32-C6, ο οποίος μέσω WiFi δίνει πρόσβαση σε λειτουργίες απομακρυσμένου ελέγχου και ενημέρωσης.

Μέσω διαφόρων αισθητήρων (φωτός, υπέρυθρων, θερμοκρασίας), ο θάλαμος μπορεί να «αντιλαμβάνεται» το περιβάλλον του και να προσαρμόζει τις λειτουργίες του αναλόγως. Η πόρτα μπορεί να λειτουργεί είτε χειροκίνητα είτε αυτόματα, με βάση την ένταση του φωτός και καθορισμένα χρονικά όρια. Αντίστοιχα, με χρήση διακοπτών μπορεί να προγραμματιστεί το τάισμα, η ενεργοποίηση προστασίας από δυσμενείς καιρικές συνθήκες και άλλες βασικές λειτουργίες.

Η LCD οθόνη 16x2 παρέχει συνεχή ενημέρωση για την κατάσταση του θαλάμου σε πραγματικό χρόνο, ενώ ταυτόχρονα, μέσω Web εφαρμογής, δίνεται δυνατότητα παρακολούθησης και παρέμβασης από οποιοδήποτε σημείο. Οι αναγνώστες RFID επιτρέπουν την καταγραφή της παρουσίας των ορνίθων εντός ή εκτός του θαλάμου, προσφέροντας ακριβή πληροφόρηση για κάθε πτηνό.

Το σύστημα ασφαλείας με ενδείξεις μέσω LED και ειδοποιήσεις στην εφαρμογή εξασφαλίζει την έγκαιρη ενημέρωση για προβλήματα όπως έλλειψη τροφής ή νερού, ή την απουσία κάποιας κότας από το θάλαμο. Τέλος, το σύστημα είναι πλήρως αυτόνομο ενεργειακά, χάρη στη χρήση μπαταρίας και φωτοβολταϊκού, καθιστώντας το κατάλληλο για απομακρυσμένες ή μη ηλεκτροδοτούμενες περιοχές.

Η εφαρμογή ενός τέτοιου συστήματος προάγει τον εκσυγχρονισμό της πτηνοτροφίας, προσφέροντας λύσεις που συνδυάζουν την τεχνολογία με την αγροτική παραγωγή, βελτιώνοντας την ποιότητα ζωής τόσο των ζώων όσο και των ανθρώπων που τα φροντίζουν.



Εικόνα 2: First Smart Coop Πηγή: smart chicken farm - Αναζήτηση Google

1.3 Οι Όρνιθες και οι Βασικές Ανάγκες Διαβίωσής τους

Οι όρνιθες (Gallus gallus domesticus) αποτελούν ένα από τα πιο διαδεδομένα οικόσιτα ζώα παγκοσμίως, προσφέροντας αυγά και κρέας, ενώ παράλληλα συμβάλλουν στη διαχείριση οργανικών αποβλήτων και στην καταπολέμηση εντόμων. Για να διασφαλιστεί η υγεία, η ευζωία και η παραγωγικότητά τους, είναι απαραίτητο να παρέχονται κατάλληλες συνθήκες διαβίωσης που καλύπτουν τις φυσιολογικές και συμπεριφορικές τους ανάγκες.

Εικόνα 3: chickens feeding Πηγή: Nutrition for Backyard Chicken Flocks - Alabama Cooperative Extension System

1.3.1 Θερμοκρασία και Περιβαλλοντικές Συνθήκες

Οι όρνιθες είναι ανθεκτικά πτηνά, αλλά ευδοκιμούν σε θερμοκρασίες μεταξύ 15°C και 24°C. Σε θερμοκρασίες κάτω των 10°C, ιδιαίτερα σε συνδυασμό με υγρασία ή ρεύματα αέρα, μπορεί να εμφανίσουν στρες, μειωμένη παραγωγή αυγών ή ακόμα και προβλήματα υγείας. Αντίστοιχα, σε υψηλές θερμοκρασίες άνω των 30°C, υπάρχει κίνδυνος θερμικής καταπόνησης. Επομένως, είναι κρίσιμο ο θάλαμος να διαθέτει κατάλληλη μόνωση, επαρκή αερισμό χωρίς ρεύματα και δυνατότητα ελέγχου της θερμοκρασίας, ώστε να διατηρούνται σταθερές και άνετες συνθήκες για τις όρνιθες καθ' όλη τη διάρκεια του έτους.

Εικόνα 4: Puffing its feathers may mean your chicken is cold Πηγή: Caring for chickens in cold weather | UMN Extension

1.3.2 Φωτισμός

Ο φωτισμός παίζει καθοριστικό ρόλο στην ωοτοκία των ορνίθων. Για να διατηρηθεί η παραγωγή αυγών σε υψηλά επίπεδα, απαιτούνται 14 έως 16 ώρες φωτός ημερησίως. Η μείωση της διάρκειας φωτισμού, ιδιαίτερα κατά τους χειμερινούς μήνες, μπορεί να οδηγήσει σε μείωση ή διακοπή της ωοτοκίας.

Η χρήση τεχνητού φωτισμού, ρυθμιζόμενου μέσω χρονοδιακοπτών, επιτρέπει τη διατήρηση σταθερού φωτισμού, προσομοιώνοντας τις φυσικές συνθήκες και υποστηρίζοντας τη συνεχή παραγωγή αυγών.

1.3.3 Διατροφή και Πρόσληψη Θρεπτικών Συστατικών

Εικόνα 5: chickens feeding Πηγή: Nutrition for Backyard Chicken Flocks - Alabama Cooperative Extension System

Η σωστή διατροφή είναι θεμελιώδης για την υγεία και την παραγωγικότητα των ορνίθων. Οι ωοτόκες όρνιθες απαιτούν τροφή με περιεκτικότητα σε πρωτεΐνη 16-18% και ασβέστιο 2,5-3,5%, απαραίτητο για τη δημιουργία σκληρών και υγιών κελυφών αυγών.

Επιπλέον, η συνεχής πρόσβαση σε καθαρό και φρέσκο νερό είναι ζωτικής σημασίας, καθώς η αφυδάτωση μπορεί να επηρεάσει αρνητικά την υγεία και την παραγωγή.

1.3.4 Χώρος και Κοινωνική Συμπεριφορά

Η παροχή επαρκούς χώρου είναι απαραίτητη για την αποφυγή στρες και επιθετικών συμπεριφορών. Συνιστάται τουλάχιστον 3-4 τετραγωνικά πόδια (0,28-0,37 τ.μ.) ανά όρνιθα εντός του θαλάμου και 10 τετραγωνικά πόδια (0,93 τ.μ.) ανά όρνιθα σε εξωτερικό χώρο. Η υπερπληθυσμιακή διαβίωση μπορεί να οδηγήσει σε επιθετικότητα, τραυματισμούς και μείωση της παραγωγής αυγών. Επιπλέον, η ύπαρξη επαρκών φωλιών και καθαρών χώρων ανάπαυσης συμβάλλει στην ευζωία των πτηνών.

Εικόνα 6: Poultry House Πηγή: HOUSING DESIGN FOR SMALL AND BACKYARD POULTRY FLOCKS – Small and backyard poultry

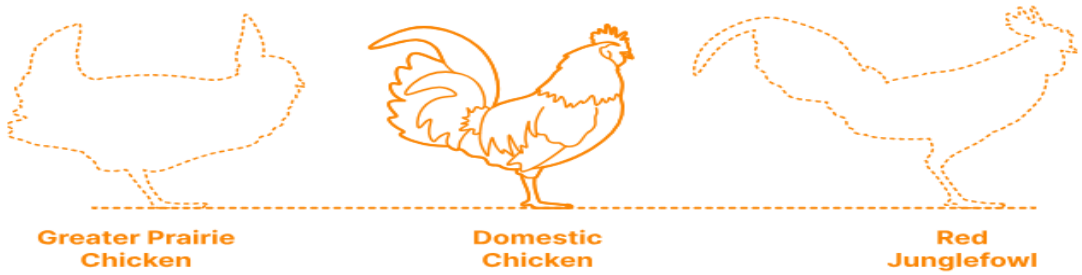
1.3.5 Ασφάλεια και Προστασία από Αρπακτικά

Οι όρνιθες είναι ευάλωτες σε επιθέσεις από αρπακτικά όπως αλεπούδες, κουνάβια και γεράκια. Η ασφαλής περίφραξη του θαλάμου, η χρήση ανθεκτικών υλικών και η αποφυγή κενών ή ανοιγμάτων είναι απαραίτητα μέτρα για την προστασία τους. Επιπλέον, η αυτόματη κλειδαριά της πόρτας κατά τις νυχτερινές ώρες ενισχύει την ασφάλεια.

1.3.6 Υγεία και Καθαριότητα

Η διατήρηση καθαρού περιβάλλοντος μειώνει τον κίνδυνο εμφάνισης ασθενειών. Ο τακτικός καθαρισμός του θαλάμου, η απομάκρυνση των περιττωμάτων και η αντικατάσταση της στρωμνής συμβάλλουν στη διατήρηση υγιεινών συνθηκών. Επιπλέον, η παρακολούθηση της συμπεριφοράς και της εμφάνισης των ορνίθων επιτρέπει την έγκαιρη ανίχνευση πιθανών προβλημάτων υγείας.

1.4 Φυσικά Χαρακτηριστικά και Καθημερινές Ανάγκες των Ορνίθων



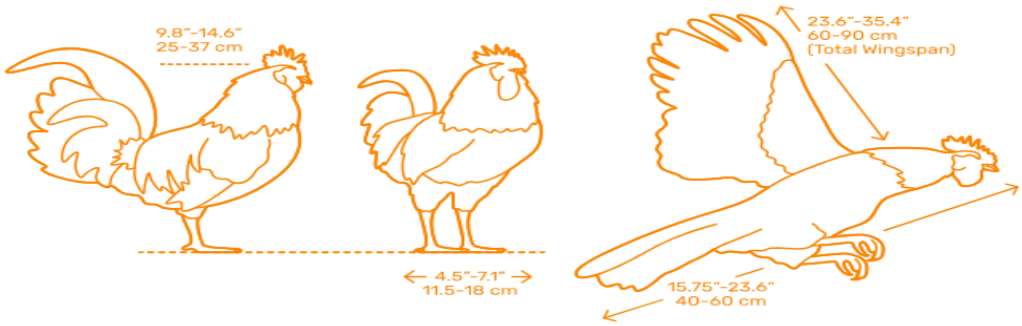
Εικόνα 7: Domestic Chicken Size Πηγή: Domestic Chicken (Gallus gallus domesticus) Dimensions & Drawings | Dimensions.com

Η επιτυχής εκτροφή των ορνίθων εξαρτάται σε μεγάλο βαθμό από την κατανόηση των φυσικών τους χαρακτηριστικών, των διατροφικών και υδρικών αναγκών τους καθώς και από τον σωστό σχεδιασμό του περιβάλλοντος διαβίωσής τους, όπως οι φωλιές.

1.4.1 Διαστάσεις και Βάρος

Οι διαστάσεις και το βάρος των ορνίθων διαφέρουν ελαφρώς ανάλογα με τη φυλή και το φύλο. Οι γενικές μέσες τιμές για ενήλικες όρνιθες είναι:

- **Μήκος σώματος**: 40–60 cm
- **Ύψος σε όρθια θέση**: 25–37 cm
- **Πλάτος σώματος**: 11,5–18 cm
- **Άνοιγμα φτερών**: 60–90 cm
- **Βάρος**: 2,6–4,5 kg



Εικόνα 8: chicken size chart Πηγή: Domestic Chicken (Gallus gallus domesticus) Dimensions & Drawings | Dimensions.com

Αυτές οι διαστάσεις είναι κρίσιμες για τη σωστή χωροθέτηση εντός του θαλάμου, ώστε να διασφαλίζεται άνεση και ελευθερία κινήσεων.

1.4.2 Καθημερινές Διατροφικές Ανάγκες

Η διατροφή των ορνίθων πρέπει να είναι ισορροπημένη για να διατηρούν την υγεία τους και να έχουν υψηλή απόδοση σε αυγά. Η ημερήσια κατανάλωση τροφής ανά ενήλικη κότα είναι:

- **Κατά μέσο όρο**: 110–120 g
- **Πρωτεΐνη**: 16–18% της συνολικής τροφής
- **Ασβέστιο**: 2,5–3,5% (για ενίσχυση του κελύφους αυγών)

Η παροχή εμπλουτισμένης τροφής και η πρόσβαση σε ασβέστιο (π.χ. τριμμένα κελύφη αυγών ή ασβεστόλιθο) είναι βασική πρακτική στις οικιακές και επαγγελματικές μονάδες.

1.4.2 Καθημερινές Υδρικές Ανάγκες

Εικόνα 9: Chicken drinks Water Πηγή: How Much Water Do Chickens Need? - Dine-A-Chook

Η επαρκής παροχή καθαρού πόσιμου νερού είναι εξίσου απαραίτητη με την τροφή. Μια μέση ενήλικη κότα χρειάζεται:

- **Νερό ημερησίως**: 250–500 ml

Η ποσότητα αυτή μπορεί να αυξηθεί κατά τη διάρκεια θερμών περιόδων ή όταν αυξάνεται η πρόσληψη πρωτεϊνών.

1.4.4 Διαστάσεις Φωλιών

Οι φωλιές είναι απαραίτητες για την ωοτοκία και πρέπει να προσφέρουν άνεση και ησυχία. Οι βασικές διαστάσεις φωλιάς για κάθε όρνιθα είναι:

- **Πλάτος**: 30–35 cm
- **Βάθος**: 30–35 cm
- **Ύψος**: 30–35 cm

Είναι καλή πρακτική να υπολογίζεται μία φωλιά για κάθε 3–4 όρνιθες. Η φωλιά πρέπει να διαθέτει απαλό υπόστρωμα και να βρίσκεται σε σημείο χαμηλού φωτισμού και θορύβου.

Εικόνα 10: chickens Nesting Πηγή: What size should my nest boxes be, and how should I place them? - My Pet Chicken

1.5 Επίλογος

Στην παρούσα ενότητα παρουσιάστηκε η μελέτη και κατασκευή ενός αυτοματοποιημένου θαλάμου για όρνιθες, με σκοπό την ευκολότερη και αποδοτικότερη φροντίδα τους, αξιοποιώντας σύγχρονες τεχνολογίες αυτοματισμού και απομακρυσμένης διαχείρισης. Η υλοποίηση βασίστηκε σε μικροελεγκτή ESP32 και περιλάμβανε αισθητήρες, διακόπτες, οθόνη LCD, RFID σύστημα, καθώς και ενεργειακή αυτονομία μέσω φωτοβολταϊκού.Η σημασία ενός τέτοιου συστήματος έγκειται τόσο στην προστασία και ευζωία των πτηνών όσο και στη διευκόλυνση του εκτροφέα, ιδιαίτερα σε συνθήκες απουσίας ή περιορισμένου χρόνου. Παράλληλα, η δυνατότητα επέκτασης και προσαρμογής του το καθιστά ιδανική λύση και για μεγαλύτερες φάρμες.

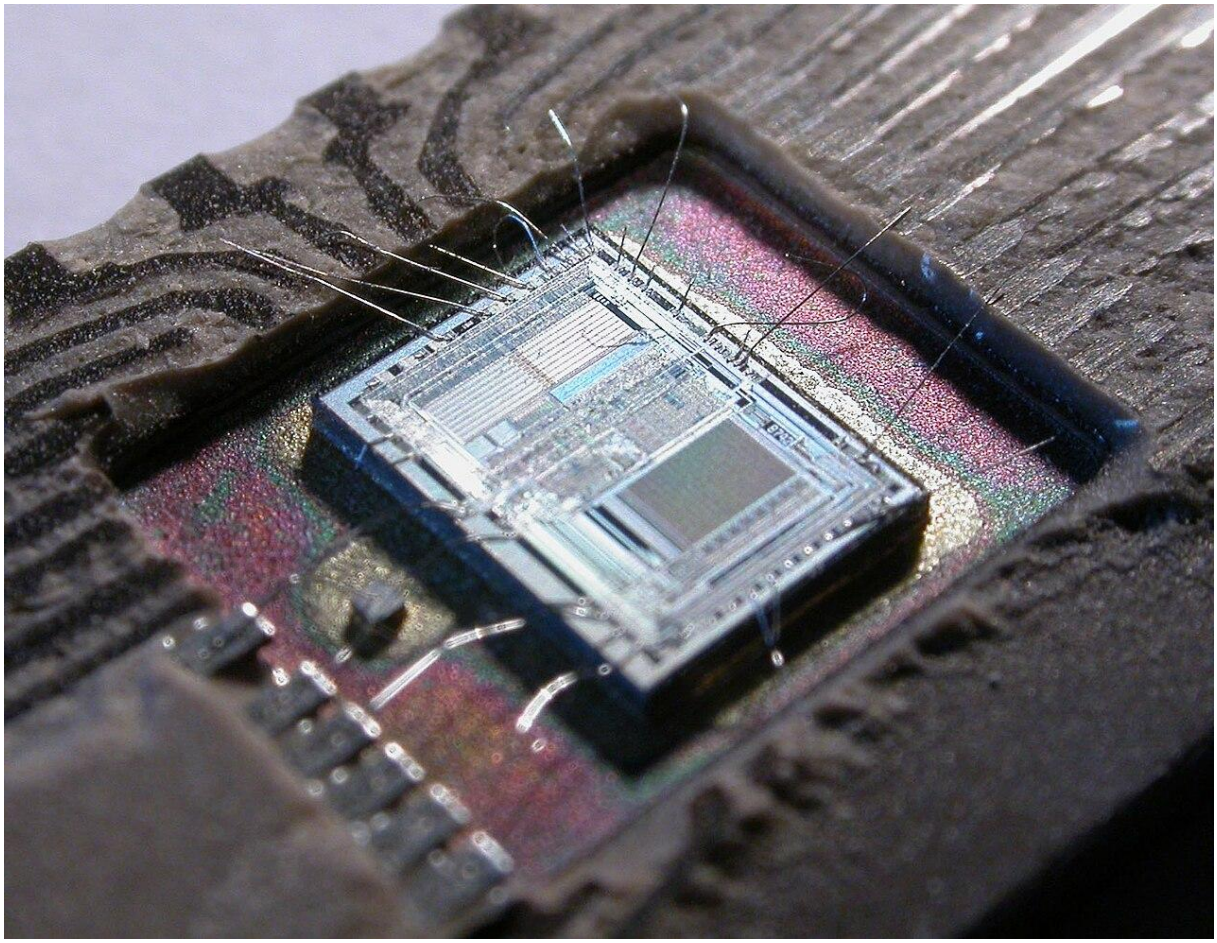
Τέλος, παρουσιάστηκαν βασικά στοιχεία σχετικά με τις ανάγκες και τα χαρακτηριστικά των ορνίθων, όπως οι απαιτήσεις σε χώρο, τροφή, νερό και οι κατάλληλες διαστάσεις φωλιών, ώστε ο σχεδιασμός του θαλάμου να καλύπτει πλήρως τις φυσιολογικές τους ανάγκες. Η ολοκληρωμένη αυτή προσέγγιση καθιστά το έργο μια πρακτική και βιώσιμη λύση για την εκτροφή πτηνών με τη βοήθεια της τεχνολογίας.

.

Κεφάλαιο 2ο: Μικροελεκτής ESP32 και τα χαρακτηριστηκά του

2.1 Εισαγωγή

Οι μικροελεγκτές (microcontrollers ή MCUs – Microcontroller Units) αποτελούν τη βάση για τον σχεδιασμό και την υλοποίηση πλήθους σύγχρονων συστημάτων αυτοματισμού, ελέγχου και έξυπνων εφαρμογών. Είναι ουσιαστικά μικροϋπολογιστές ενσωματωμένοι σε ένα μόνο chip, που διαθέτουν CPU, μνήμη (RAM και ROM/Flash), και περιφερειακά εισόδου/εξόδου (I/O).

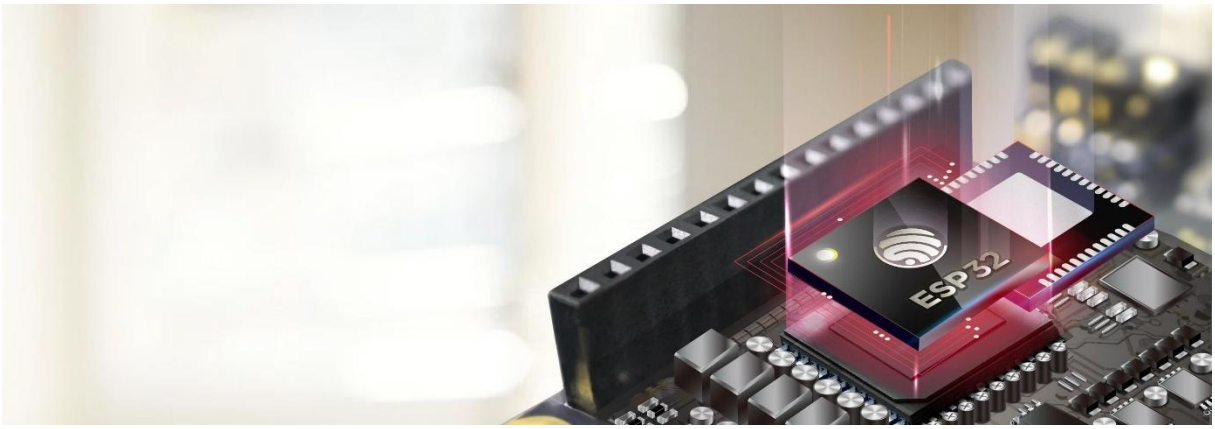


Εικόνα 11: Intel MCU Πηγή:www.wikipedia.com

Χρησιμοποιούνται ευρέως σε οικιακές συσκευές, ιατρικά μηχανήματα, βιομηχανικά συστήματα, ρομπότ, αυτοκίνητα, αλλά και σε καινοτόμες DIY εφαρμογές, όπως και στην παρούσα εργασία. Η βασική τους λειτουργία είναι να εκτελούν συγκεκριμένες εργασίες σε πραγματικό χρόνο, με χαμηλή κατανάλωση ενέργειας, αξιοπιστία και επεκτασιμότητα.

Ο προγραμματισμός τους γίνεται συνήθως με γλώσσες όπως η C/C++ ή Python (π.χ. με χρήση MicroPython), ενώ πολλά μοντέλα υποστηρίζονται από ευέλικτα περιβάλλοντα ανάπτυξης όπως το Arduino IDE, το PlatformIO ή το επίσημο ESP-IDF για προϊόντα της Espressif.

2.2 Ο Μικροελεγκτής ESP32:Γενικά



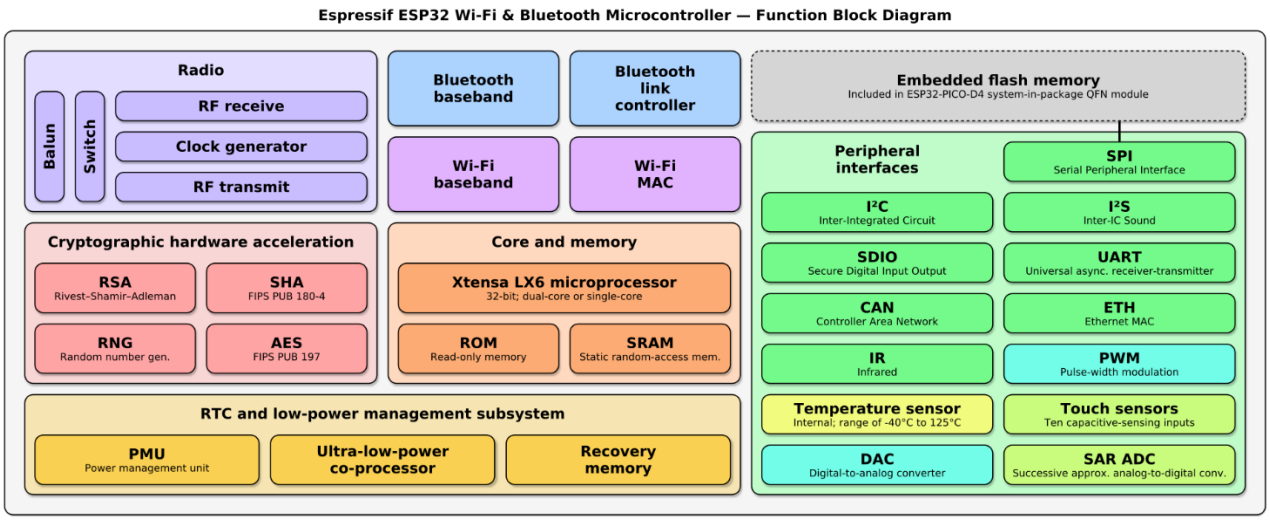
Εικόνα 12: ESP32 Πηγή: https://www.espressif.com/

Ο ESP32 είναι ένας ισχυρός και ευέλικτος μικροελεγκτής που σχεδιάστηκε από την εταιρεία Espressif Systems. Πρόκειται για έναν από τους πιο δημοφιλείς μικροελεγκτές που χρησιμοποιούνται σε εφαρμογές IoT (Internet of Things), αυτοματισμών, έξυπνων κατοικιών, ρομποτικής και άλλων embedded συστημάτων, χάρη στις υψηλές επιδόσεις του, την ενσωματωμένη ασύρματη συνδεσιμότητα και το χαμηλό κόστος.

Η αρχιτεκτονική του ESP32 επιτρέπει την αυτόνομη λειτουργία συσκευών, αφού συνδυάζει σε ένα μόνο chip όλες τις απαραίτητες υπομονάδες: επεξεργαστή, μνήμη, είσοδο/έξοδο και δυνατότητα επικοινωνίας

2.2.1 Επεξεργαστική Ισχύς και Πυρήνες

Ο ESP32 βασίζεται στον διπύρηνο επεξεργαστή Xtensa LX6 της Cadence, με συχνότητα λειτουργίας που μπορεί να φτάσει έως και τα 240 MHz. Κάθε πυρήνας μπορεί να λειτουργεί ανεξάρτητα ή συνεργατικά, κάτι που δίνει στον προγραμματιστή τη δυνατότητα να αφιερώσει έναν πυρήνα για real-time λειτουργίες (όπως ανάγνωση αισθητήρων ή διαχείριση διακοπτών) και τον άλλο για δευτερεύουσες εργασίες όπως η ασύρματη επικοινωνία ή ο χειρισμός ενός Web Server.



Εικόνα 13: ESP32 Block Diagram Πηγή: https://grobotronics.com/images/companies/1/esp32_technical_reference_manual_en.pdf?1623656830214

Ο επεξεργαστής υποστηρίζει επίσης λειτουργίες εξοικονόμησης ενέργειας (Dynamic Frequency Scaling), κάτι ιδιαίτερα χρήσιμο για φορητές ή αυτόνομες εφαρμογές που τροφοδοτούνται με μπαταρίες ή φωτοβολταϊκά..

2.2.2 Μνήμη RAM και Flash

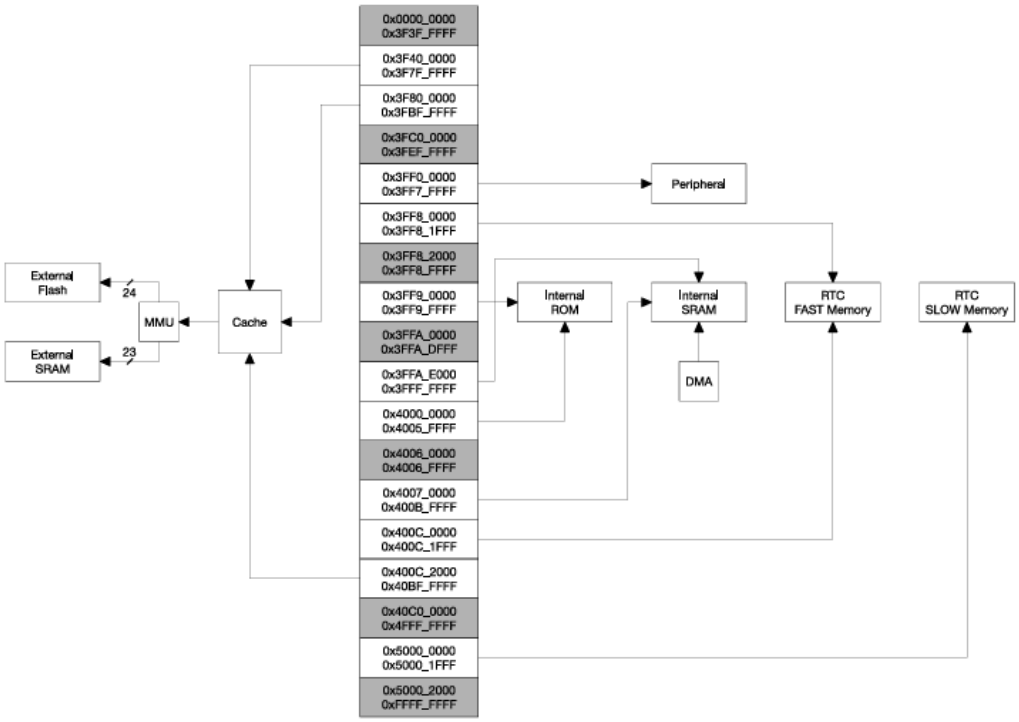


Figure 2: System Address Mapping

Εικόνα 14: ESP32 System Address Mapping Πηγή https://grobotronics.com/images/companies/1/esp32_technical_reference_manual_en.pdf?1623656830214

Ο ESP32 διαθέτει 520 KB SRAM και υποστηρίζει εξωτερική Flash μνήμη έως και 16 MB (τυπικά 4 MB σε έτοιμες πλακέτες, όπως η ESP32 DevKit ή WROOM). Η RAM χρησιμοποιείται για την προσωρινή αποθήκευση δεδομένων κατά την εκτέλεση του προγράμματος, ενώ η Flash χρησιμοποιείται για την αποθήκευση του ίδιου του λογισμικού (firmware).

Υπάρχει επίσης RTC (Real Time Clock) memory για διατήρηση δεδομένων κατά την κατάσταση ύπνου.

2.2.3 Επικοινωνία και Ασύρματη Συνδεσιμότητα

Εικόνα 15: ESP32C6 Πηγή:https://www.grobotronics.gr

Ένα από τα μεγαλύτερα πλεονεκτήματα του ESP32 είναι η **ενσωματωμένη υποστήριξη για Wi-Fi και Bluetooth**, χωρίς να χρειάζονται εξωτερικά modules:

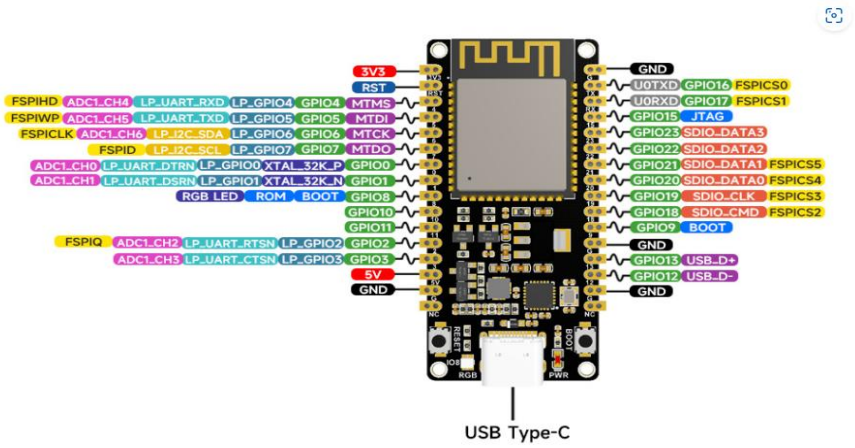
- **Wi-Fi 802.11 b/g/n**: Λειτουργεί στα 2.4 GHz και μπορεί να λειτουργεί είτε ως client είτε ως access point, ή ακόμα και ταυτόχρονα (softAP + STA).
- **Bluetooth v4.2**: Υποστηρίζει **Bluetooth Classic** και **Bluetooth Low Energy (BLE)**, ιδανικό για επικοινωνία με κινητές συσκευές ή άλλες χαμηλής κατανάλωσης συσκευές.

Με αυτόν τον τρόπο, ο μικροελεγκτής μπορεί να επικοινωνεί με τοπικά δίκτυα ή το Internet, να φιλοξενεί **web servers**, να στέλνει ειδοποιήσεις, ή να δέχεται εντολές από απομακρυσμένους χρήστες.

2.2.4 Θύρες Εισόδου/Εξόδου (GPIO και Περιφερειακά)

Ο ESP32 διαθέτει περισσότερα από **30 ψηφιακά και αναλογικά GPIO pins** (ανάλογα με τη διάταξη της πλακέτας), τα οποία μπορούν να διαμορφωθούν ως:

- **Ψηφιακές Είσοδοι/Εξοδοι (Digital I/O)**
- **Αναλογικές Είσοδοι (ADC – 12-bit)**: μέχρι 18 κανάλια
- **Αναλογικές Έξοδοι (DAC – 8-bit)**: 2 κανάλια
- **PWM Outputs**: Για έλεγχο ταχύτητας μοτέρ, φωτεινότητας LED κ.λπ.
- **UART, SPI, I2C, CAN, I2S**: Πλήθος πρωτοκόλλων για επικοινωνία με αισθητήρες, οθόνες, RFID, GPS, SD cards, κ.λπ.



Εικόνα 16: ESP32 pin Out Πηγή: https://grobotronics.com/waveshare-esp32-c6-developement-board-esp32-c6-dev-kit-n8-el.html

Η δυνατότητα αυτής της πληθώρας διεπαφών επιτρέπει τη διασύνδεση του ESP32 με πλήθος εξωτερικών περιφερειακών, καθιστώντας τον εξαιρετικά ευέλικτο.

2.2.5 Χαμηλή Κατανάλωση Ενέργειας

Ο ESP32 έχει σχεδιαστεί με γνώμονα την **ενεργειακή απόδοση**. Διαθέτει διάφορες λειτουργίες εξοικονόμησης ενέργειας, όπως:

- **Light Sleep**
- **Deep Sleep**
- **Ultra Low Power (ULP) co-processor**

Εικόνα 17: Low Power Figure Πηγή: https://www.espressif.com

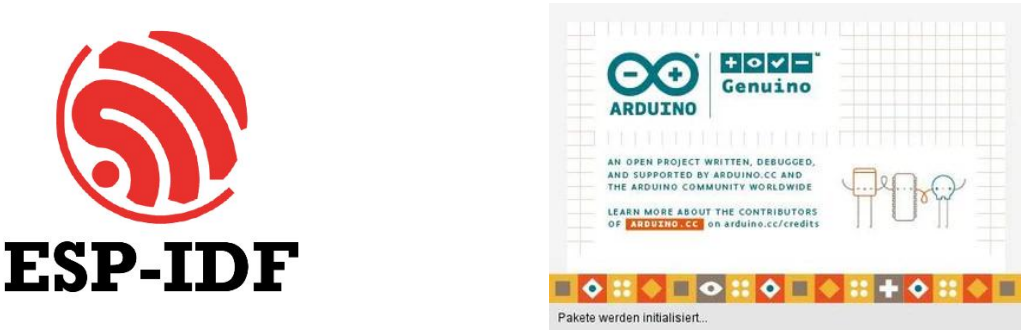
Αυτές οι λειτουργίες είναι κρίσιμες σε συστήματα που λειτουργούν με **μπαταρία ή φωτοβολταϊκά**, όπως στην παρούσα εργασία.

2.2.6 Ανάπτυξη και Προγραμματισμός

Ο ESP32 υποστηρίζεται από πολλές δημοφιλείς πλατφόρμες προγραμματισμού:

- **Arduino IDE** (με τις κατάλληλες βιβλιοθήκες ESP32)
- **PlatformIO**
- **ESP-IDF (Espressif IoT Development Framework)** – το επίσημο SDK
- **MicroPython** και **Lua**

Η ευκολία χρήσης του Arduino IDE τον καθιστά προσβάσιμο σε αρχάριους, ενώ το ESP-IDF επιτρέπει πιο επαγγελματική ανάπτυξη για απαιτητικά projects.



Εικόνα 18: ESP IDF & ARDUINO IDE Πηγή: esp idf - Αναζήτηση Google

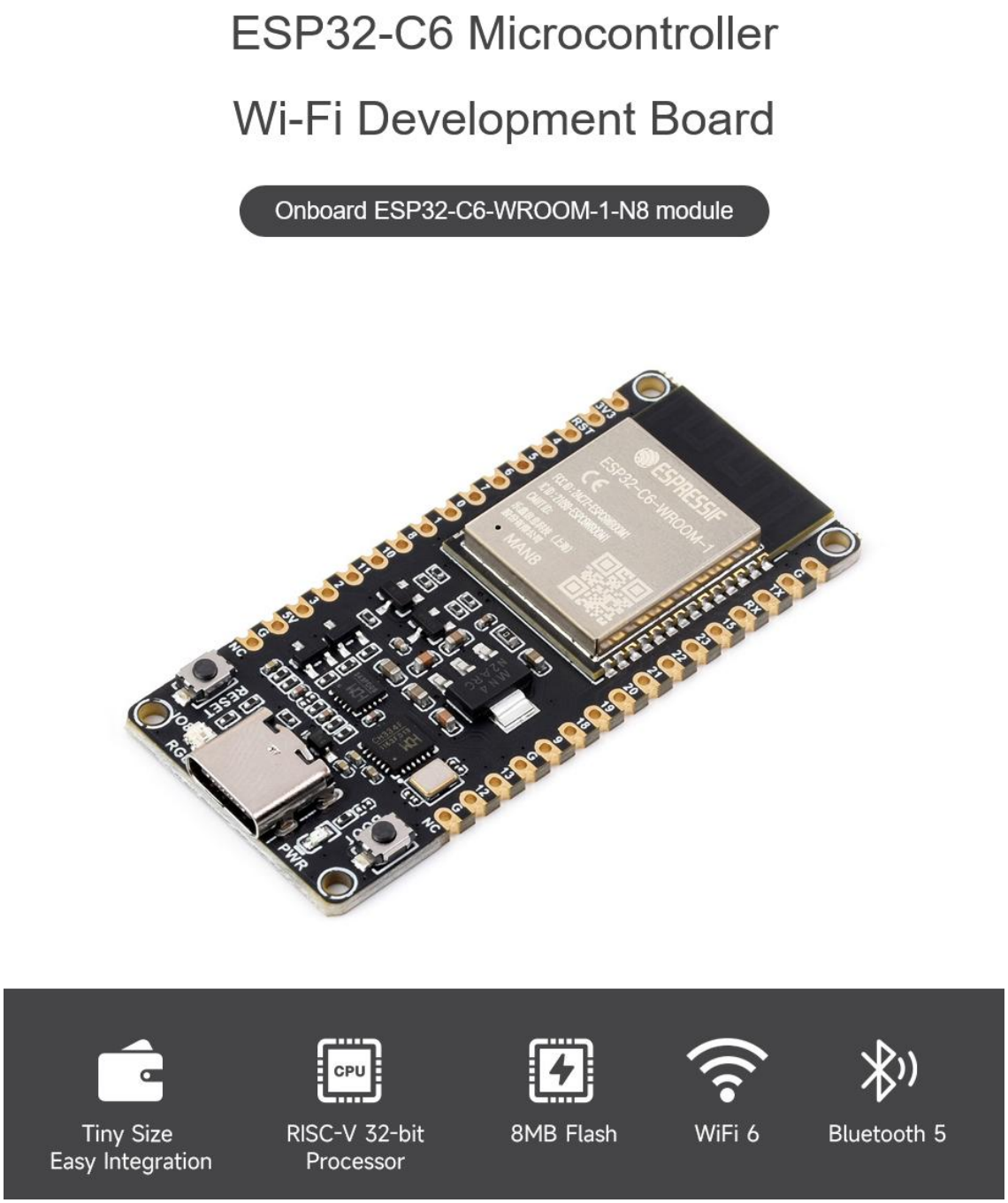
2.2.7 Πλεονεκτήματα και Εφαρμογές

Ο ESP32 αποτελεί μία από τις **πληρέστερες και πιο αποδοτικές λύσεις** στην αγορά μικροελεγκτών, ιδανική για:

- Έξυπνες οικιακές συσκευές (home automation)
- Συστήματα συναγερμού / παρακολούθησης
- Συστήματα παρακολούθησης περιβάλλοντος
- Ρομποτική
- Wearables
- Smart farming (όπως ο αυτόματος θάλαμος ορνίθων της παρούσας εργασίας)

Το γεγονός ότι προσφέρει **Wi-Fi, Bluetooth, επεξεργαστική ισχύ, επεκτασιμότητα και οικονομία**, τον καθιστά τον «χρυσό κανόνα» για τους σύγχρονους αυτοματισμούς και IoT εφαρμογές.

2.3 ESP32-C6 DevKit N8: Εξειδικευμένα Χαρακτηριστικά και Εφαρμογή



Εικόνα 19: ESP32C6 Πηγή:https://www.grobotronics.gr

Ο **ESP32-C6 DevKit N8** αποτελεί μια αναβαθμισμένη και σύγχρονη έκδοση της σειράς ESP32 και σχεδιάστηκε με γνώμονα τις σύγχρονες απαιτήσεις του **Internet of Things (IoT)**. Βασισμένος στο νέο chip **ESP32-C6**, προσφέρει σημαντικές βελτιώσεις τόσο σε επίπεδο απόδοσης όσο και συνδεσιμότητας σε σχέση με τους προκατόχους του, ενσωματώνοντας τεχνολογίες όπως **Wi-Fi 6** και **Bluetooth 5.0 Low Energy**, καθώς και χαμηλότερη κατανάλωση ενέργειας.Η πλακέτα **ESP32-C6 DevKit N8** είναι το development board που χρησιμοποιείται στην παρούσα εργασία ως «εγκέφαλος» του αυτοματοποιημένου θαλάμου. Παρακάτω παρουσιάζονται τα βασικά τεχνικά και λειτουργικά του χαρακτηριστικά:

2.3.1 Κύρια Τεχνικά Χαρακτηριστικά

Επεξεργαστής: RISC-V 32-bit single-core, έως 160 MHz

- Ο ESP32-C6 χρησιμοποιεί έναν **RISC-V πυρήνα**, που προσφέρει αποδοτικότητα και σύγχρονη αρχιτεκτονική με έμφαση στην ενεργειακή οικονομία.

Μνήμη:

- **512 KB SRAM**
- **8 MB Flash** (εξ ου και το "N8" στην ονομασία του development board)

Ασύρματη συνδεσιμότητα:

- **Wi-Fi 6 (802.11ax)**
  - Προσφέρει μεγαλύτερη σταθερότητα, ταχύτερες ταχύτητες και χαμηλότερη καθυστέρηση.
  - Υποστηρίζει OFDMA για ταυτόχρονη επικοινωνία με πολλαπλές συσκευές.
- **Bluetooth 5.0 LE**
  - Ιδανικό για χαμηλής κατανάλωσης ασύρματη επικοινωνία, π.χ. με αισθητήρες, κινητά ή wearable συσκευές.

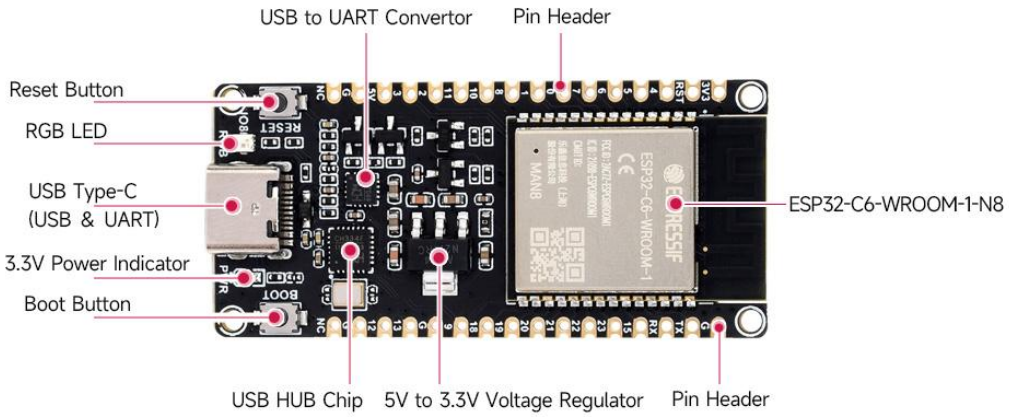
Περιφερειακά και Είσοδοι/Εξοδοι:

- Μέχρι **23 GPIO pins**
- **2 ADCs**, 12-bit, για σύνδεση αισθητήρων αναλογικών τιμών (π.χ. θερμοκρασίας, φωτός)
- **PWM, SPI, I2C, UART**, και άλλες διεπαφές για επικοινωνία με εξωτερικά υποσυστήματα

Υποστήριξη ασφαλούς επικοινωνίας: Secure Boot, Flash Encryption, Hardware Cryptography

Κατανάλωση ενέργειας:

- Πολύ χαμηλή κατανάλωση σε κατάσταση ύπνου (**Deep Sleep <5 μA**), καθιστώντας τον κατάλληλο για αυτόνομα συστήματα με μπαταρία ή φωτοβολταϊκά.

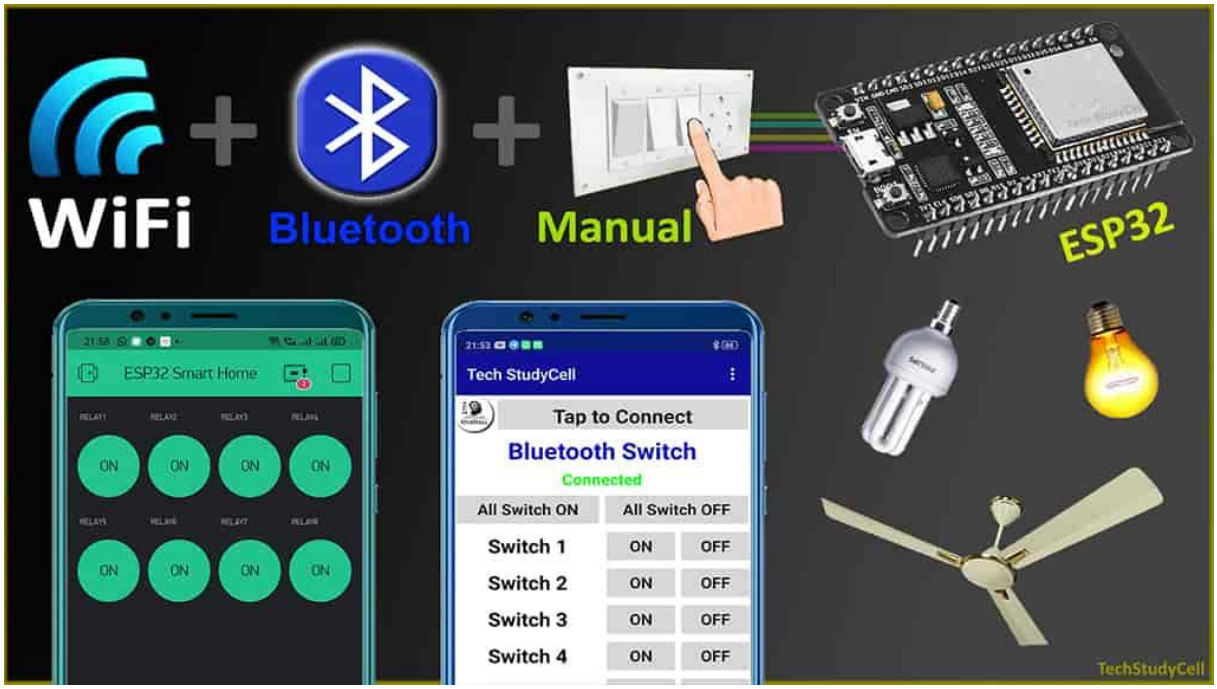


Εικόνα 20: ESP32C6 DEV KIT Πηγή:https://www.grobotronics.gr

2.3.2 Λόγοι Επιλογής στην Παρούσα Εργασία

Ο ESP32-C6 επιλέχθηκε για την υλοποίηση της πτυχιακής εργασίας για τους εξής βασικούς λόγους:

- **Αυξημένη ενεργειακή απόδοση:** Ιδανικός για ένα αυτόνομο σύστημα που λειτουργεί με φωτοβολταϊκή ενέργεια και μπαταρία.
- **Wi-Fi 6 για σταθερή σύνδεση και απομακρυσμένο έλεγχο:** Επιτρέπει τον έλεγχο του θαλάμου μέσω Web εφαρμογής, από3333 οποιοδήποτε σημείο του κόσμου.
- **Πλήθος GPIO και υποστηριζόμενων πρωτοκόλλων:** Διευκολύνει τη σύνδεση με αισθητήρες φωτός, θερμοκρασίας, RFID, μοτέρ για την πόρτα, ταΐστρες και LED.
- **Υποστήριξη Bluetooth 5.0 LE:** Δυνατότητα μελλοντικής επέκτασης του συστήματος για ασύρματη επικοινωνία με κινητές συσκευές ή wearable ετικέτες στα πτηνά.
- **Μικρό μέγεθος και χαμηλό κόστος:** Πολύ σημαντικά πλεονεκτήματα για μια οικονομική αλλά αξιόπιστη κατασκευή.



Εικόνα 21: ESP32 Communication Πηγή: esp32 smart picture - Αναζήτηση Google

2.4 Επίλογος Κεφαλαίου

Στο παρόν κεφάλαιο παρουσιάστηκε το γενικό πλαίσιο των μικροελεγκτών, η λειτουργία τους και ο ρόλος τους σε συστήματα αυτοματισμού. Ιδιαίτερη έμφαση δόθηκε στον **ESP32-C6 DevKit N8**, τον οποίο χρησιμοποιεί η εργασία ως βασική μονάδα ελέγχου του αυτοματοποιημένου θαλάμου με όρνιθες. Τα τεχνικά του χαρακτηριστικά, όπως η υποστήριξη **Wi-Fi 6**, η χαμηλή κατανάλωση ενέργειας και η πληθώρα εισόδων/εξόδων, τον καθιστούν ιδανικό για απομακρυσμένο έλεγχο και πλήρως αυτόνομη λειτουργία. Η επιλογή του συγκεκριμένου μικροελεγκτή βασίστηκε τόσο στις τεχνικές του δυνατότητες όσο και στη σταθερότητα που προσφέρει, επιβεβαιώνοντας την καταλληλότητά του για εφαρμογές στον τομέα του IoT και της σύγχρονης γεωργίας ή κτηνοτροφίας.

Κεφάλαιο 3ο: Αισθητήρες, Περιφερειακά και Συνδεσιμότητα του Συστήματος

3.1 Εισαγωγή

Για τη δημιουργία ενός έξυπνου και πλήρως αυτοματοποιημένου θαλάμου με όρνιθες, η χρήση του μικροελεγκτή από μόνη της δεν είναι αρκετή. Ο πυρήνας του συστήματος ενισχύεται ουσιαστικά από ένα πλήθος αισθητήρων και περιφερειακών συσκευών που επιτρέπουν στο σύστημα να "αντιλαμβάνεται" το περιβάλλον του, να αλληλεπιδρά με αυτό, και να λαμβάνει έξυπνες αποφάσεις.

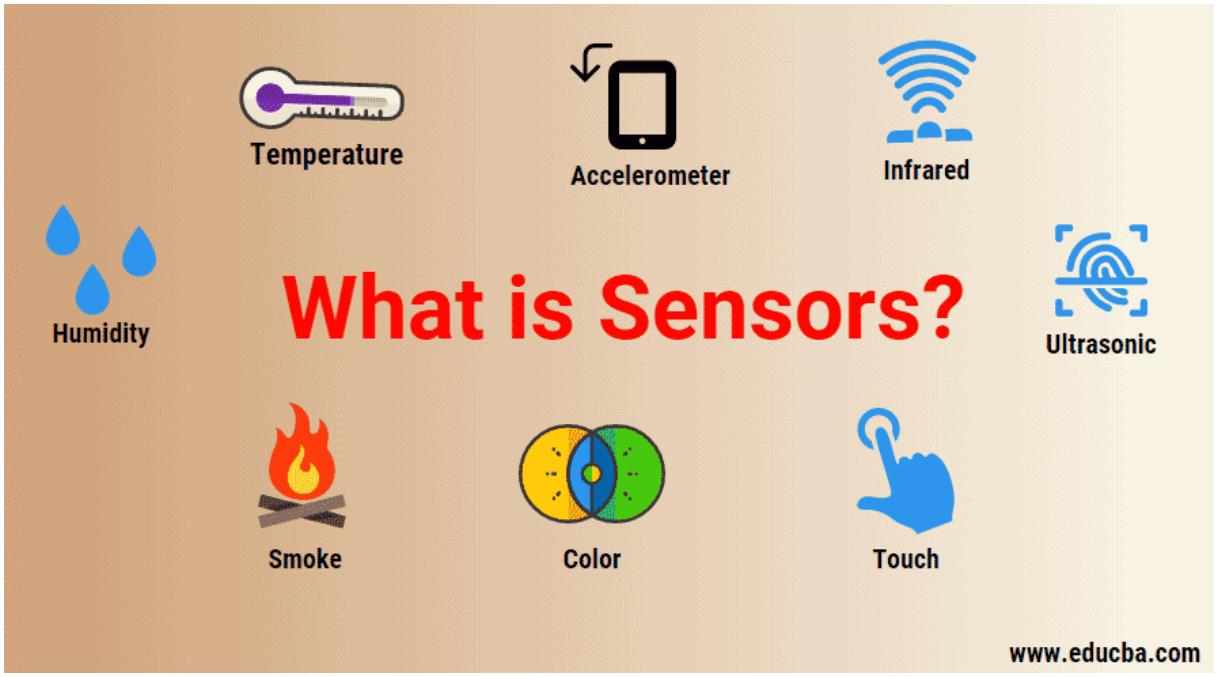
Οι αισθητήρες προσφέρουν δεδομένα για κρίσιμες παραμέτρους, όπως η θερμοκρασία, η παρουσία βροχής, το επίπεδο φωτός ή η κίνηση των ζώων. Με βάση αυτά τα δεδομένα, το σύστημα μπορεί να ενεργοποιήσει ή να απενεργοποιήσει μηχανισμούς, να στείλει ειδοποιήσεις ή να προσαρμόσει τη λειτουργία του με αυτονομία και ακρίβεια. Παράλληλα, οι περιφερειακές συσκευές όπως οθόνες, κινητήρες, RFID αναγνώστες και χρονόμετρα προσφέρουν τη δυνατότητα πληροφόρησης του χρήστη και φυσικής αλληλεπίδρασης με το περιβάλλον του θαλάμου.

Επιπλέον, η συνδεσιμότητα αποτελεί θεμέλιο λίθο της εξυπνάδας του συστήματος, επιτρέποντας την απομακρυσμένη παρακολούθηση και επέμβαση μέσω διαδικτύου, με στόχο τη διαρκή εποπτεία και επέκταση της λειτουργικότητας.

Στο παρόν κεφάλαιο παρουσιάζονται αναλυτικά όλα τα υλικά που χρησιμοποιήθηκαν για την κατασκευή του αυτόματου θαλάμου. Για κάθε αισθητήρα ή περιφερειακή μονάδα, αναλύεται η βασική αρχή λειτουργίας του, ο τρόπος ενσωμάτωσής του στο σύστημα και η συνεισφορά του στη συνολική λειτουργία της εφαρμογής.

3.2 Αισθητήρες (Sensors)

Οι αισθητήρες αποτελούν τις «αισθήσεις» ενός αυτοματοποιημένου συστήματος, αφού προσφέρουν τη δυνατότητα στο σύστημα να λαμβάνει δεδομένα από το περιβάλλον και να τα επεξεργάζεται με στόχο τη λήψη έξυπνων και προσαρμοσμένων αποφάσεων. Ανάλογα με τον τύπο τους, οι αισθητήρες καταγράφουν φυσικά φαινόμενα όπως θερμοκρασία, φως, υγρασία ή κίνηση και μετατρέπουν αυτές τις πληροφορίες σε ηλεκτρικά σήματα που μπορεί να επεξεργαστεί ο μικροελεγκτής.
Στο παρόν σύστημα, οι αισθητήρες διαχωρίζονται σε τρεις βασικές υποκατηγορίες: **αισθητήρες περιβάλλοντος**, οι οποίοι παρέχουν δεδομένα για τις καιρικές και φωτεινές συνθήκες· **αισθητήρες κίνησης/παρουσίας**, που εντοπίζουν την παρουσία ή απουσία των ορνίθων· και **συστήματα αναγνώρισης RFID**, που προσφέρουν προσωποποιημένο εντοπισμό κάθε πτηνού ξεχωριστά, μέσω ειδικών αναγνωριστικών. Ο σωστός συνδυασμός και η αξιοποίηση αυτών των αισθητήρων είναι καθοριστικής σημασίας για την αξιοπιστία και την αποδοτικότητα του αυτόματου θαλάμου.



Εικόνα 22: What Is Sensor Πηγή: sensors - Αναζήτηση Google

3.2.1 Αισθητήρες Περιβάλλοντος

Οι αισθητήρες περιβάλλοντος συλλέγουν συνεχώς δεδομένα για φυσικές παραμέτρους του θαλάμου, όπως η θερμοκρασία, η παρουσία βροχής ή χιονιού και το επίπεδο φωτισμού. Αυτές οι μετρήσεις είναι καθοριστικής σημασίας για τη διατήρηση συνθηκών ευζωίας των ορνίθων: εξασφαλίζουν ότι ο θάλαμος δεν εκτίθεται σε υπερβολική ζέστη ή παγετό, ότι οι πόρτες παραμένουν κλειστές όταν βρέχει, και ότι η λειτουργία της πόρτας συντονίζεται με την ημέρα και τη νύχτα. Με τη βοήθεια των θερμίστορ, των αισθητήρων βροχής και των φωτοαντιστάσεων, το σύστημα προσαρμόζεται αυτόματα στις μεταβαλλόμενες καιρικές και φωτιστικές συνθήκες, χωρίς ανθρώπινη παρέμβαση.

Εικόνα 23: Environment Sensor Πηγή: sensors - Αναζήτηση Google

3.2.1.1 NTC Θερμίστορ 10K 3950 (μέτρηση θερμοκρασίας)

Εικόνα 24: NTC Thermistor Πηγή:https://www.grobotronics.gr

Το **θερμίστορ** (thermistor) είναι ένας τύπος θερμικού αισθητήρα που αλλάζει την **αντίστασή του ανάλογα με τη θερμοκρασία**. Στην περίπτωση του **NTC (Negative Temperature Coefficient)** θερμίστορ, η **αντίσταση μειώνεται καθώς αυξάνεται η θερμοκρασία**.

Το συγκεκριμένο μοντέλο είναι τύπου **NTC 10K 3950**, που σημαίνει:

- **10K**: η αντίσταση είναι 10.000 Ohm στους 25°C.
- **3950**: σταθερά B (B-constant), που δείχνει την ευαισθησία του αισθητήρα στις αλλαγές θερμοκρασίας.
- **Γενικά χαρακτηριστικά:**
- **Τύπος αισθητήρα**: Παθητικός, χρειάζεται κύκλωμα ανάγνωσης με αναλογική είσοδο.
- **Θερμοκρασιακό εύρος λειτουργίας**: Συνήθως από -50°C έως +125°C.
- **Μορφή**: Είναι επικαλυμμένο με εποξική ρητίνη (epoxy), που το καθιστά αδιάβροχο και κατάλληλο για χρήση σε υγρά ή υγρά περιβάλλοντα.
- **Σύνδεση**: Περιλαμβάνει καλώδιο για εύκολη σύνδεση σε breadboard ή μικροελεγκτές.

Πώς λειτουργεί:

Το θερμίστορ λειτουργεί ως **μεταβλητή αντίσταση**. Όταν η θερμοκρασία αλλάζει, αλλάζει και η αντίστασή του. Αυτή η μεταβολή μετατρέπεται σε αναλογικό σήμα (τάση) από τον μικροελεγκτή (όπως ο ESP32), που στη συνέχεια ερμηνεύεται με βάση μαθηματικές συναρτήσεις (π.χ. εξίσωση Steinhart-Hart ή look-up tables).

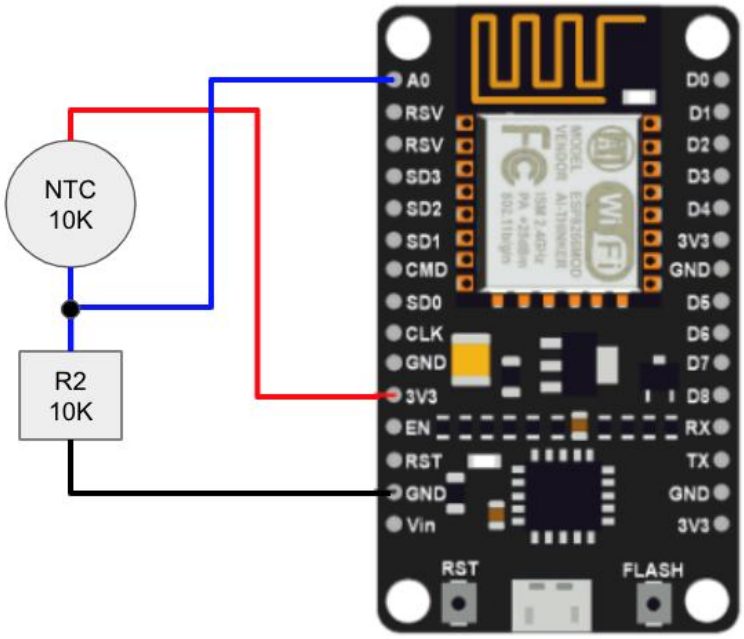
Γενική χρήση:

- Μέτρηση θερμοκρασίας σε συστήματα HVAC, ενυδρεία, συσκευές κουζίνας, αυτοκινητοβιομηχανία.
- Παρακολούθηση υπερθέρμανσης σε ηλεκτρονικά κυκλώματα.
- Ανίχνευση και έλεγχος περιβαλλοντικής θερμοκρασίας σε αυτοματισμούς.
- **Ρόλος στην πτυχιακή εργασία:**

Στον **αυτοματοποιημένο θάλαμο με όρνιθες**, το θερμίστορ NTC 10K 3950 χρησιμοποιείται για να:

- **Παρακολουθεί τη θερμοκρασία** στο εσωτερικό του θαλάμου, εξασφαλίζοντας κατάλληλες συνθήκες για τις όρνιθες.
- **Ενεργοποιεί ή απενεργοποιεί** λειτουργίες όπως η αυτόματη πόρτα, εφόσον η θερμοκρασία πέσει κάτω από ένα ασφαλές όριο (π.χ. παγετός).
- Παρέχει **δεδομένα σε πραγματικό χρόνο** στην LCD οθόνη ή μέσω της Web εφαρμογής.
- Συνεισφέρει στη **λειτουργία ασφάλειας**, ενημερώνοντας τον χρήστη σε περίπτωση επικίνδυνης θερμοκρασιακής μεταβολής.

Η επιλογή ενός τέτοιου απλού αλλά αξιόπιστου αισθητήρα είναι ιδανική για low-power εφαρμογές και αγροτικά περιβάλλοντα.



Εικόνα 25: NTC Basic Connection Πηγή: ntc thermistor with esp32 - Αναζήτηση Google

3.2.1.2 MH-RD Αισθητήρας Βροχής/Χιονιού

Εικόνα 26: MH-RD Rain sensor Πηγή:https://www.devobox.gr

Ο **MH-RD** είναι ένας αισθητήρας περιβαλλοντικής υγρασίας επιφανείας, σχεδιασμένος για την **ανίχνευση σταγονιδίων νερού**, όπως βροχής, χιονιού ή υγρασίας, πάνω σε μια μεταλλική επιφάνεια. Χρησιμοποιείται ευρέως σε **συστήματα αυτοματισμού**, μετεωρολογικούς σταθμούς και εφαρμογές όπου είναι σημαντικό να γνωρίζουμε αν υπάρχει βροχόπτωση ή γενικά παρουσία νερού.

Γενικά χαρακτηριστικά:

- **Σύνθεση**: Περιλαμβάνει μία **μεταλλική επιφάνεια (Rain Board)** με αγώγιμες διαδρομές και ένα κύκλωμα ελέγχου (module).
- **Τρόπος λειτουργίας**: Όταν πέσει νερό στην επιφάνεια, δημιουργείται αγώγιμη σύνδεση μεταξύ των γραμμών, μεταβάλλοντας την ηλεκτρική αντίσταση.

- **Έξοδοι**:
    - **Αναλογική έξοδος** (AO): Παρέχει επίπεδο τάσης που σχετίζεται με την ποσότητα υγρασίας.
    - **Ψηφιακή έξοδος** (DO): Ενεργοποιείται/απενεργοποιείται όταν ξεπεραστεί ένα προκαθορισμένο όριο υγρασίας.
- **Ρυθμιζόμενο όριο**: Μέσω ενσωματωμένου ποτενσιόμετρου στο module.

Πώς λειτουργεί:

Το κύκλωμα λειτουργεί ως αισθητήρας αγωγιμότητας. Σε ξηρή επιφάνεια, η αντίσταση είναι υψηλή και δεν ανιχνεύεται παρουσία νερού. Μόλις το νερό έρθει σε επαφή με την επιφάνεια, η αντίσταση μειώνεται και ο αισθητήρας παράγει έξοδο που μπορεί να αναγνωρίσει ένας μικροελεγκτής, όπως ο ESP32.

Γενική χρήση:

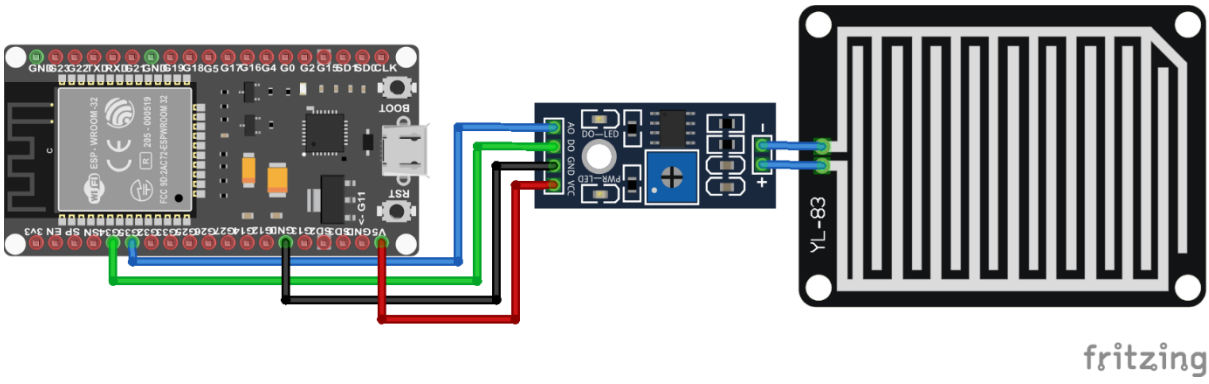
- Συστήματα **έξυπνου ποτίσματος** (μην ποτίσεις αν βρέχει).
- **Αυτόματες τέντες**, πατζούρια ή φώτα.
- **Ασφάλεια παραθύρων και στεγανότητας**.
- **Ανίχνευση παγετού**, σε συνδυασμό με θερμοκρασία.

Ρόλος στην πτυχιακή εργασία:

Ο MH-RD έχει βασική σημασία στο **σύστημα αυτοματοποιημένου θαλάμου με όρνιθες**:

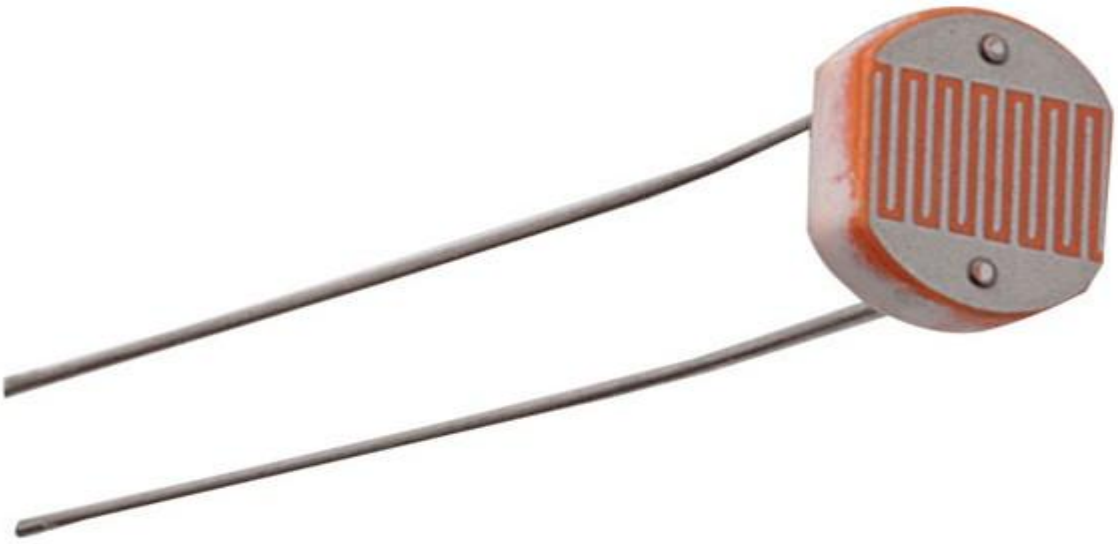
- Ανιχνεύει **βροχή ή χιόνι**, προκειμένου να κρατηθεί **κλειστή η πόρτα** του θαλάμου για την προστασία των πτηνών.
- Συμβάλλει στο **σενάριο ασφαλείας** σε συνδυασμό με το θερμίστορ, αποτρέποντας την έκθεση των ορνίθων σε ακραίες καιρικές συνθήκες.
- Στέλνει **ειδοποιήσεις μέσω της web εφαρμογής** σε περίπτωση συνεχούς παρουσίας νερού ή ανωμαλίας.

Είναι απλός στη χρήση, οικονομικός και αξιόπιστος για περιβαλλοντική παρακολούθηση.



Εικόνα 27: Rain Sensor Basic connection with ESP Πηγή: rain sensor esp - Αναζήτηση Google

3.2.1.3 Φωτοαντίσταση LDR 5mm (μέτρηση φωτεινότητας)



Εικόνα 28: LDR 5mm Πηγή:https://www.grbotronics.gr

Η **LDR** (Light Dependent Resistor) ή αλλιώς **φωτοαντίσταση** είναι ένας **παθητικός αισθητήρας φωτός** που αλλάζει την **ηλεκτρική του αντίσταση ανάλογα με την ένταση του φωτός** που πέφτει πάνω του.

Γενικά χαρακτηριστικά:

- **Διάμετρος**: 5mm (συνηθισμένο μέγεθος για εφαρμογές με μικροελεγκτές).
- **Υλικό**: Φωτοευαίσθητα ημιαγώγιμα υλικά, συνήθως χαλκοθειούχο κάδμιο (CdS).
- **Αντίσταση σε σκοτάδι**: Πολύ υψηλή (π.χ. >1 ΜΩ).
- **Αντίσταση σε έντονο φως**: Πολύ χαμηλή (π.χ. <1 ΚΩ).
- **Είδος εξόδου**: Αναλογική – απαιτεί ανάγνωση από αναλογική είσοδο (ADC).

Πώς λειτουργεί:

Η LDR λειτουργεί με βάση την **φωτοαγωγιμότητα**. Όταν το φως προσπίπτει στο αισθητήριο υλικό, **μειώνεται η αντίστασή του**. Ο μικροελεγκτής μπορεί να μετρήσει αυτήν την αλλαγή και να εκτιμήσει το επίπεδο φωτεινότητας του περιβάλλοντος.

Η LDR δεν παράγει από μόνη της σήμα – πρέπει να χρησιμοποιηθεί σε **διαιρέτη τάσης**, ώστε οι αλλαγές στην αντίστασή της να οδηγήσουν σε αντίστοιχη αλλαγή της τάσης που διαβάζει ο ESP32.

Γενική χρήση:

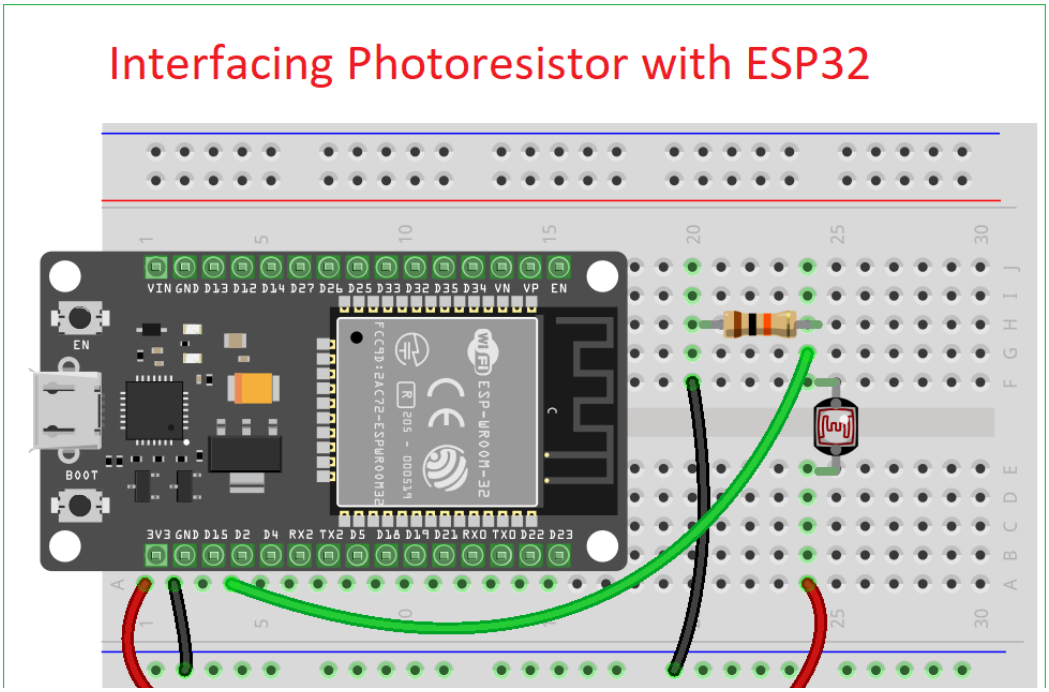
- **Αυτόματος φωτισμός** (ανάβει φώτα το βράδυ).
- **Φωτορυθμικά συστήματα**.
- **Ασφάλεια – Ανίχνευση σκιάς ή παρουσίας**.
- **Αισθητήρες φωτός σε κινητά ή κάμερες**.
- **Ηλιακά συστήματα** για βέλτιστο προσανατολισμό.

Ρόλος στην πτυχιακή εργασία:

Στον **αυτοματοποιημένο θάλαμο με όρνιθες**, η LDR χρησιμοποιείται κυρίως για τον:

- **Αυτόματο έλεγχο της πόρτας**: Η πόρτα του κοτετσιού **ανοίγει όταν ανιχνευθεί φως** (ξημέρωμα) και **κλείνει όταν σκοτεινιάσει** (ηλιοβασίλεμα).
- Εξασφαλίζει ότι οι **όρνιθες δεν μένουν εκτεθειμένες** τη νύχτα.
- Βοηθά στη δημιουργία **αυτόνομου και "έξυπνου" ελέγχου** χωρίς την ανάγκη καθημερινής παρέμβασης του χρήστη.

Η LDR είναι ιδιαίτερα **αξιόπιστη, φθηνή και εύκολη στην εγκατάσταση**, καθιστώντας την ιδανική για αγροτικές και απομακρυσμένες εφαρμογές όπως η παρούσα.



Εικόνα 29: LDR Basic connection with ESP Πηγή: photoresistor esp32 - Αναζήτηση Google

3.2.2 Αισθητήρες Κίνησης/Παρουσίας

Οι αισθητήρες κίνησης και παρουσίας εντοπίζουν με ακρίβεια τη διέλευση ή την παραμονή των ορνίθων στον θάλαμο. Χρησιμοποιώντας τεχνολογίες υπέρυθρης δέσμης (IR break beam) και μαγνητικούς διακόπτες (door/window sensors), το σύστημα μπορεί να καταμετρά ποια πτηνά εισέρχονται ή εξέρχονται, να ελέγχει ότι η πόρτα έχει κλείσει σωστά μετά από κάθε διέλευση, και να στέλνει άμεσα ειδοποιήσεις σε περίπτωση μη αναμενόμενης κίνησης ή παραμονής εκτός θαλάμου. Με αυτό τον τρόπο, διατηρείται ασφαλές περιβάλλον για τις κότες και αποτρέπονται απώλειες.

3.2.2.1 Αισθητήρας Υπέρυθρης Δέσμης (IR Break Beam)

Εικόνα 30: IR Break Beam Sensor Πηγή:https://www.grobotronics.gr

Ο **IR Break Beam** είναι ένας **αισθητήρας ανίχνευσης παρουσίας ή διέλευσης** που λειτουργεί με βάση τη **διακοπή μιας υπέρυθρης ακτίνας** μεταξύ δύο στοιχείων: ενός **πομπού** (IR LED) και ενός **δέκτη** (φωτοτρανζίστορ ή IR φωτοδιόδος). Ανάμεσα τους σχηματίζεται μια αόρατη δέσμη υπερύθρων.

Γενικά χαρακτηριστικά:

- **Μέγεθος LED**: 3mm (κατάλληλο για στενά σημεία εγκατάστασης).
- **Απόσταση λειτουργίας**: Συνήθως 10-30cm, ανάλογα με τις συνθήκες.
- **Τάση λειτουργίας**: 3.3V – 5V.
- **Ψηφιακή έξοδος**: Ενεργοποιείται όταν **διακοπεί η ακτίνα**.

Πώς λειτουργεί:

Ο πομπός στέλνει συνεχώς υπέρυθρη ακτινοβολία προς τον δέκτη. Όσο αυτή η δέσμη παραμένει **ανενόχλητη**, το σύστημα παραμένει «ήσυχο». Όταν όμως **κάτι (π.χ. ένα αντικείμενο ή ζώο) περάσει ανάμεσα** στους δύο, η δέσμη **διακόπτεται**, οπότε το κύκλωμα αναγνωρίζει την παρουσία.

Γενική χρήση:

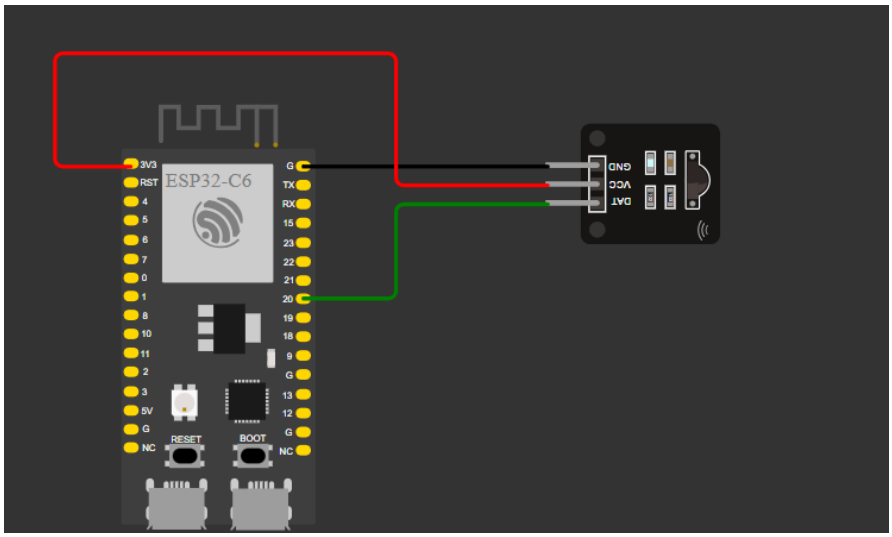
- **Ανίχνευση εισόδου/εξόδου** σε πόρτες.
- **Συστήματα ασφαλείας**.
- **Καταμέτρηση αντικειμένων ή ανθρώπων**.
- **Αυτόματες πόρτες**, μηχανισμοί κίνησης.
- **Ρομποτικές εφαρμογές** για εμπόδια.

Ρόλος στην πτυχιακή εργασία:

Στο **αυτοματοποιημένο κοτέτσι**, ο IR Break Beam χρησιμοποιείται για:

- **Ανίχνευση ποσότητας φαγητού** ένα ζεύγος αισθητήρων βρίσκεται μέσα στην ταΐστρα οπότε σε περίπτωση που η στάθμη του φαγητού πέσει κάτω από ένα επίπεδο η ακτίνα του αισθητήρα επανενωνεται και έτσι έχουμε ειδοποίηση για χαμηλή ποσότητα φαγητού.

Ο αισθητήρας είναι αξιόπιστος, δεν επηρεάζεται από το φυσικό φως (σε αντίθεση με τις LDR) και λειτουργεί χωρίς επαφή, πράγμα ιδανικό για περιβάλλον με ζώα.



Εικόνα 31: ESP32 Connection With IR Break Beam

3.2.2.2 Αισθητήρας Πόρτας/Παραθύρου (MC-38W) (κατάσταση πόρτας)

Εικόνα 32: MC-38 Πηγή: Anga MC-38W Αισθητήρας Πόρτας/Παραθύρου Βιδωτή και Αυτοκόλλητη Ευρείας Χρήσης σε Λευκό Χρώμα 10τμχ 650-021 | Skroutz.gr

Ο **Anga MC-38W** είναι ένας απλός αλλά αξιόπιστος **μαγνητικός διακόπτης επαφής (reed switch)**, σχεδιασμένος για να ανιχνεύει το **άνοιγμα και το κλείσιμο** πόρτας, παραθύρου ή οποιασδήποτε κινούμενης επιφάνειας. Αποτελείται από **δύο κομμάτια** – έναν μαγνήτη και έναν αισθητήρα (μαγνητικό διακόπτη) που τοποθετούνται αντικριστά.

Γενικά χαρακτηριστικά:

- **Τύπος διακόπτη**: Reed switch (μαγνητικός).
- **Τάση λειτουργίας**: έως 100V DC.
- **Ρεύμα λειτουργίας**: έως 0.5A.
- **Μοντάζ**: Με **βίδες ή αυτοκόλλητο**, κατάλληλο για ξύλινες/πλαστικές επιφάνειες.
- **Απλή συνδεσμολογία**: Δύο καλώδια – on/off επαφή.

Πώς λειτουργεί:

Όταν ο μαγνήτης βρίσκεται **κοντά στον διακόπτη**, η μαγνητική δύναμη **κρατάει κλειστή την επαφή**. Αν απομακρυνθεί (δηλαδή αν ανοίξει η πόρτα), η επαφή **ανοίγει** και το κύκλωμα ενημερώνεται για την αλλαγή κατάστασης. Η λειτουργία του βασίζεται σε **μηχανισμό χωρίς επαφή και χωρίς τριβή**, οπότε έχει μεγάλη διάρκεια ζωής.

Γενική χρήση:

- **Συστήματα συναγερμού** σε πόρτες/παράθυρα.
- **Έξυπνες κατοικίες** για παρακολούθηση κίνησης.
- **Αυτόματα φώτα** (άναμμα όταν ανοίγει πόρτα).
- **Καταγραφή κινήσεων** σε βιομηχανικές εφαρμογές.

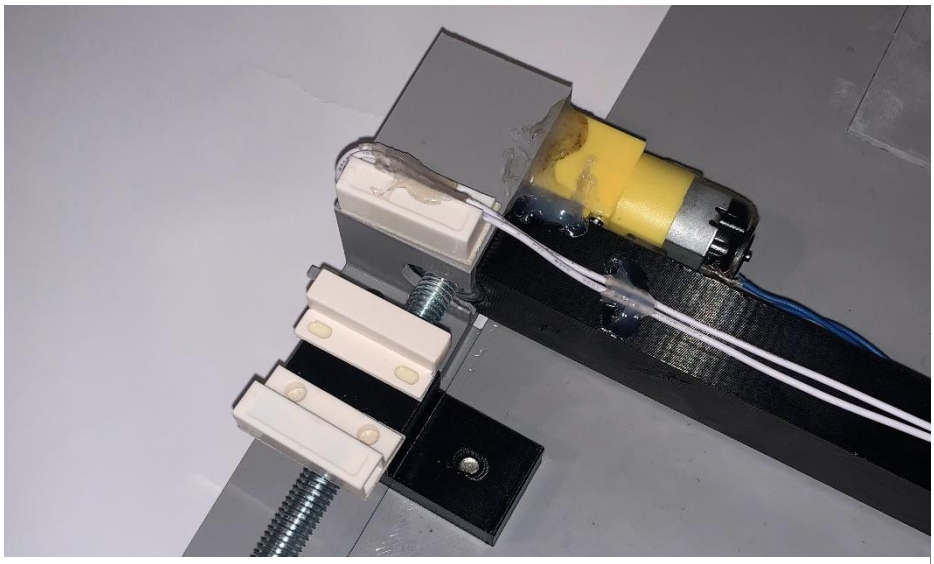
Ρόλος στην πτυχιακή εργασία:

Στον **αυτοματοποιημένο θάλαμο με όρνιθες**, ο αισθητήρας Anga MC-38W τοποθετείται στην **πόρτα του κοτετσιού** και παρέχει:

- **Έλεγχο θέσης** της πόρτας: γνωρίζουμε αν είναι **ανοιχτή ή κλειστή** με ακρίβεια.
- **Επαλήθευση σωστής λειτουργίας** του αυτοματισμού (σε συνδυασμό με τον κινητήρα).
- **Ενεργοποίηση ειδοποίησης** στην εφαρμογή σε περίπτωση που η πόρτα **δεν έχει κλείσει σωστά** ή έχει κολλήσει λόγω εμποδίου ή βλάβης.

Έτσι έχοντας δυο τέτοιους αισθητήρες ένα στο πάνω σημείο της πόρτας και έναν στο κάτω μπορούμε να ελέγξουμε που βρίσκεται η πόρτα καθώς και για το πότε να σταματήσει το μοτέρ που είναι υπεύθυνο για να ανέβει και κατέβει η πόρτα

Ο αισθητήρας αυτός είναι πολύ **χαμηλού κόστους**, **χωρίς ανάγκη τροφοδοσίας**, και με **απλή εγκατάσταση**, καθιστώντας τον ιδανικό για εξωτερικές αγροτικές εφαρμογές.

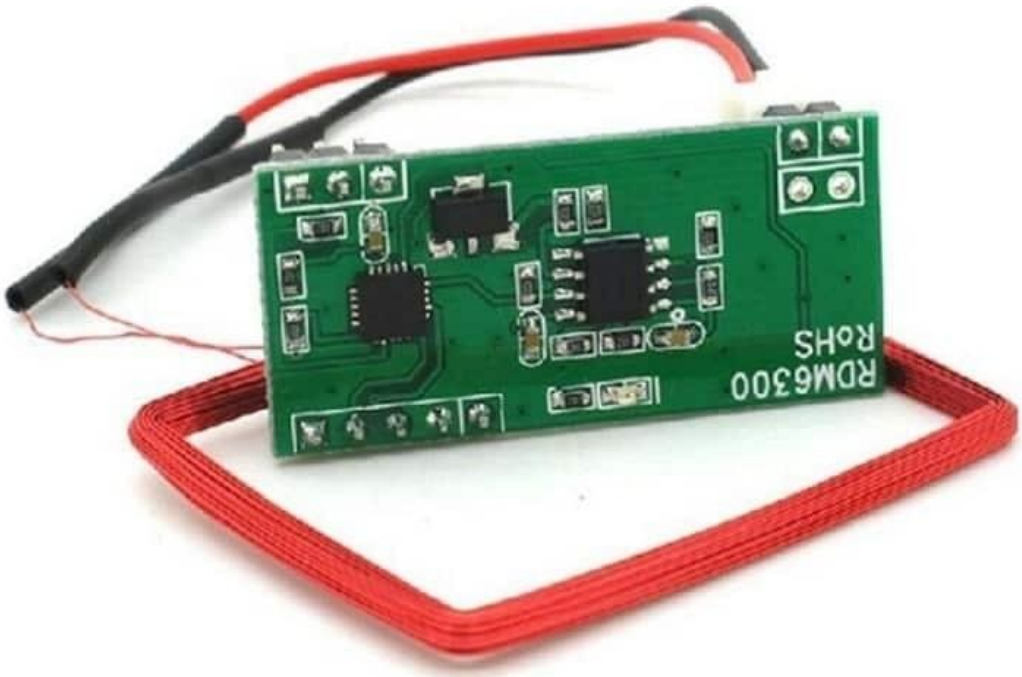


Εικόνα 33: Εφαρμογή στη πόρτα του θαλάμου

3.2.3 Αναγνώριση RFID

Το σύστημα αναγνώρισης RFID προσδίδει στο κοτέτσι «ηλεκτρονική ταυτότητα» σε κάθε κότα, μέσω παθητικών RFID μπρελόκ που φέρουν μοναδικό αναγνωριστικό. Ο αναγνώστης RDM6300 διαβάζει κάθε tag κατά τη διέλευση, και ο μικροελεγκτής καταγράφει ποια κότα πέρασε, σε ποια ώρα, καθώς και την κατάσταση του θαλάμου. Αυτή η ακριβής καταγραφή επιτρέπει τη δημιουργία εξατομικευμένων προφίλ κάθε πτηνού, διευκολύνει την παρακολούθηση της υγείας και της συμπεριφοράς τους, και εξασφαλίζει ότι καμία κότα δεν χάνεται ή διαφεύγει της προσοχής.

3.2.3.1 RFID Reader RDM6300 UART 125kHz



Εικόνα 34: RDM6300 Πηγή:https://www.devobox.gr

Ο **RDM6300** είναι ένας ευρέως διαδεδομένος αναγνώστης RFID τεχνολογίας 125kHz, σχεδιασμένος για χρήση με παθητικά RFID tags. Λειτουργεί μέσω σειριακής επικοινωνίας UART, γεγονός που τον καθιστά εξαιρετικά συμβατό με μικροελεγκτές όπως ο ESP32, διευκολύνοντας την ανάγνωση μοναδικών αναγνωριστικών από tags και μπρελόκ.

Γενικά Χαρακτηριστικά:

- **Τάση λειτουργίας:** 3.3V ή 5V
- **Συχνότητα:** 125kHz
- **Διασύνδεση:** UART (TX/RX)
- **Απόσταση ανάγνωσης:** 2 – 10 cm ανάλογα με το tag
- **Ταχύτητα μετάδοσης:** 9600bps
- **Υποστήριξη tags:** EM4100 ή συμβατά UNIQUE 125kHz

Τρόπος Λειτουργίας:

Ο αναγνώστης εκπέμπει ένα χαμηλής συχνότητας ηλεκτρομαγνητικό πεδίο (125kHz), το οποίο ενεργοποιεί τα παθητικά RFID tags. Κατά την επαφή ή προσέγγιση ενός tag, ο RDM6300 αναγνωρίζει το μοναδικό 64-bit ID του και το στέλνει ως ακολουθία ASCII χαρακτήρων μέσω της σειριακής εξόδου.

Η σύνδεσή του με τον μικροελεγκτή είναι ιδιαίτερα απλή, απαιτώντας μόνο τροφοδοσία και τις γραμμές RX/TX. Δεν χρειάζεται καμία σύνθετη ρύθμιση, ενώ η απουσία εξωτερικής κεραίας τον καθιστά συμπαγή και εύχρηστο.

Χρήση σε μικροελεγκτικά έργα:

Ο RDM6300 βρίσκει εφαρμογή σε πλήθος έργων ελέγχου πρόσβασης, αυτοματισμών, ή παρακολούθησης παρουσίας. Είναι κατάλληλος για συστήματα όπου απαιτείται χαμηλό κόστος, αξιοπιστία και ευκολία στην ενσωμάτωση.

Ρόλος στην παρούσα εργασία:

Στο αυτόματο κοτέτσι, ο RDM6300 λειτουργεί ως το βασικό μέσο **αναγνώρισης των ορνίθων**, μέσω μοναδικών μπρελόκ RFID που προσαρμόζονται σε κάθε πτηνό. Κατά τη διέλευση από την είσοδο του θαλάμου, ο αναγνώστης καταγράφει την ταυτότητα της κότας, και το σύστημα ενημερώνει τη βάση δεδομένων για το ποια πτηνά είναι παρόντα. Έτσι, επιτυγχάνεται **ηλεκτρονική παρακολούθηση** και **ασφάλεια**, καθώς και δυνατότητα **καταμέτρησης και παρακολούθησης συμπεριφοράς** κάθε ζώου ξεχωριστά.

3.2.3.2 RFID Tags/Μπρελόκ 125kHz UNIQUE

Εικόνα 35: RFID Tag Πηγή:https://www.grobotronics.gr

Τα **RFID Tags** τύπου μπρελόκ 125kHz είναι παθητικές συσκευές που φέρουν ένα ενσωματωμένο κύκλωμα το οποίο περιέχει έναν μοναδικό αναγνωριστικό αριθμό (ID). Λειτουργούν χωρίς μπαταρία και ενεργοποιούνται από το μαγνητικό πεδίο που δημιουργεί ο RFID αναγνώστης (όπως ο RDM6300).

Γενικά Χαρακτηριστικά:

- **Τύπος:** Παθητικό RFID μπρελόκ
- **Συχνότητα λειτουργίας:** 125kHz
- **Τεχνολογία:** EM4100 / UNIQUE format (64-bit ID)
- **Τροφοδοσία:** Δεν απαιτείται (παθητικό tag)
- **Απόσταση λειτουργίας:** 2 – 10 cm
- **Διάρκεια ζωής:** Πολύ μεγάλη (δεν φθείρεται εύκολα)
- **Μορφή:** Πλαστικό μπρελόκ με κρίκο

Τρόπος Λειτουργίας:

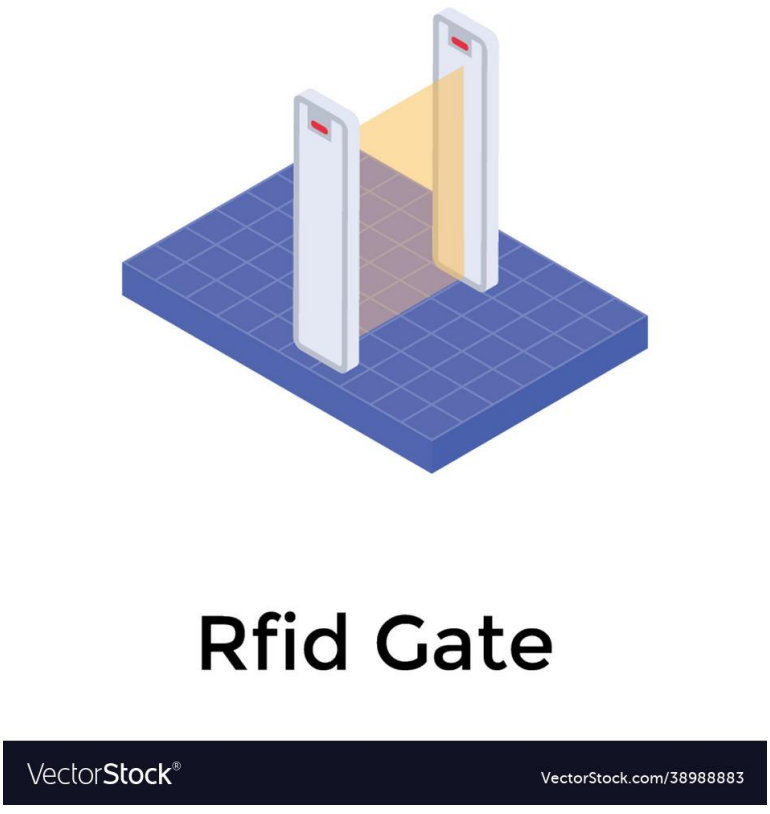
Κατά την προσέγγιση σε έναν ενεργοποιημένο RFID reader, το tag απορροφά ενέργεια από το παραγόμενο ηλεκτρομαγνητικό πεδίο, ενεργοποιείται προσωρινά, και στέλνει το μοναδικό του ID πίσω στον αναγνώστη. Αυτό το ID μπορεί να χρησιμοποιηθεί για αναγνώριση ή καταγραφή παρουσίας.

Χρήση σε μικροελεγκτικά έργα:

Τα μπρελόκ RFID χρησιμοποιούνται σε εφαρμογές όπως ηλεκτρονικά κλειδιά, ελεγχόμενη πρόσβαση, ταυτοποίηση ζώων, tracking συσκευών, ακόμη και σε έργα με Arduino για τη δημιουργία "έξυπνων θυρών" ή συστημάτων παρακολούθησης.

Ρόλος στην παρούσα εργασία:

Στο έξυπνο κοτέτσι, κάθε κότα φέρει **το δικό της RFID tag** προσαρτημένο στο σώμα της με ασφαλή τρόπο . Κατά τη διέλευσή της από την πόρτα, το tag διαβάζεται από έναν από τους RDM6300, επιτρέποντας στο σύστημα να γνωρίζει **ποια κότα μπήκε ή βγήκε**, και αν λείπει κάποιο ζώο. Αυτό παρέχει **ποσοτική και ποιοτική εποπτεία** του πληθυσμού των ορνίθων σε πραγματικό χρόνο.



Εικόνα 36: RFID Gate System Πηγή: rfid gate scketch - Αναζήτηση Google

3.3 Περιφερειακά και Ενεργά Μέρη

Σε ένα αυτοματοποιημένο σύστημα, όπως αυτό που αναπτύχθηκε στην παρούσα εργασία, οι αισθητήρες αποτελούν μεν τα "μάτια και τα αυτιά" του συστήματος, όμως τα **περιφερειακά και ενεργά μέρη** είναι εκείνα που επιτρέπουν την **έμπρακτη αλληλεπίδραση** με το περιβάλλον. Αυτά τα εξαρτήματα ενεργοποιούν ή απενεργοποιούν λειτουργίες, παρουσιάζουν πληροφορίες στον χρήστη και δίνουν στον μικροελεγκτή τη δυνατότητα να **επιδρά υλικά** στον φυσικό κόσμο. Οθόνες, κινητήρες, οδηγοί ισχύος και συστήματα εισόδου/εξόδου είναι μερικά από τα βασικά περιφερειακά που καθιστούν το σύστημα λειτουργικό και χρηστικό στην πράξη.

3.3.1 Οθόνες και Συστήματα Χρονισμού

Οι οθόνες και τα συστήματα χρονισμού αποτελούν βασικά περιφερειακά για την παροχή πληροφορίας και το συγχρονισμό λειτουργιών σε ένα έξυπνο σύστημα. Οι οθόνες, όπως οι LCD χαρακτήρων, επιτρέπουν την άμεση οπτική παρακολούθηση κρίσιμων παραμέτρων και καταστάσεων από τον χρήστη, χωρίς την ανάγκη χρήσης εφαρμογών ή υπολογιστών. Ταυτόχρονα, τα συστήματα χρονισμού, όπως το RTC (Real-Time Clock), διασφαλίζουν την ακρίβεια στην καταγραφή και εκτέλεση γεγονότων που εξαρτώνται από τον χρόνο. Η συνδυασμένη χρήση τους ενισχύει τη διαφάνεια, τον έλεγχο και την αξιοπιστία του αυτόματου θαλάμου.

Εικόνα 37: Different Displays Πηγή: electronics project displays - Αναζήτηση Google

3.3.1.1 Basic 16x2 Character LCD - White on Blue 5V

Εικόνα 38: LCD 16X2 Πηγή:https://www.grobotronics.gr

Η **LCD 16x2** είναι μία από τις πιο διαδεδομένες και δημοφιλείς οθόνες χαρακτήρων που χρησιμοποιούνται σε συστήματα με μικροελεγκτές, όπως Arduino, ESP32 και άλλα. Ο αριθμός **16x2** δηλώνει ότι η οθόνη μπορεί να εμφανίσει **16 χαρακτήρες σε 2 γραμμές**, σύνολο δηλαδή 32 χαρακτήρες.

Γενικά Χαρακτηριστικά:

- **Τύπος**: Οθόνη Υγρών Κρυστάλλων (Liquid Crystal Display - LCD)
- **Ανάλυση**: 16 στήλες x 2 γραμμές
- **Τάση λειτουργίας**: 5V (κάποιες παραλλαγές μπορούν να λειτουργήσουν και με 3.3V)
- **Κατανάλωση**: Πολύ χαμηλή, ιδανική για αυτόνομα συστήματα
- **Οπίσθιος φωτισμός**: Συνήθως έχει μπλε φόντο με λευκούς χαρακτήρες, αν και υπάρχουν πολλές εκδοχές (π.χ. πράσινο με μαύρους χαρακτήρες)

Τρόπος Λειτουργίας:

Η LCD 16x2 βασίζεται στο **HD44780 controller**, ένα δημοφιλές chip που διευκολύνει την επικοινωνία με μικροελεγκτές. Η σύνδεσή της μπορεί να γίνει:

- **Παραδοσιακά** με 8 ή 4 καλώδια δεδομένων + έλεγχο (RS, E, RW)
- **Μέσω I2C interface**, το οποίο απλοποιεί σημαντικά τη συνδεσμολογία, μειώνοντας τις απαραίτητες συνδέσεις μόνο σε 2 καλώδια δεδομένων (SDA & SCL)

Χρήση σε μικροελεγκτικά έργα:

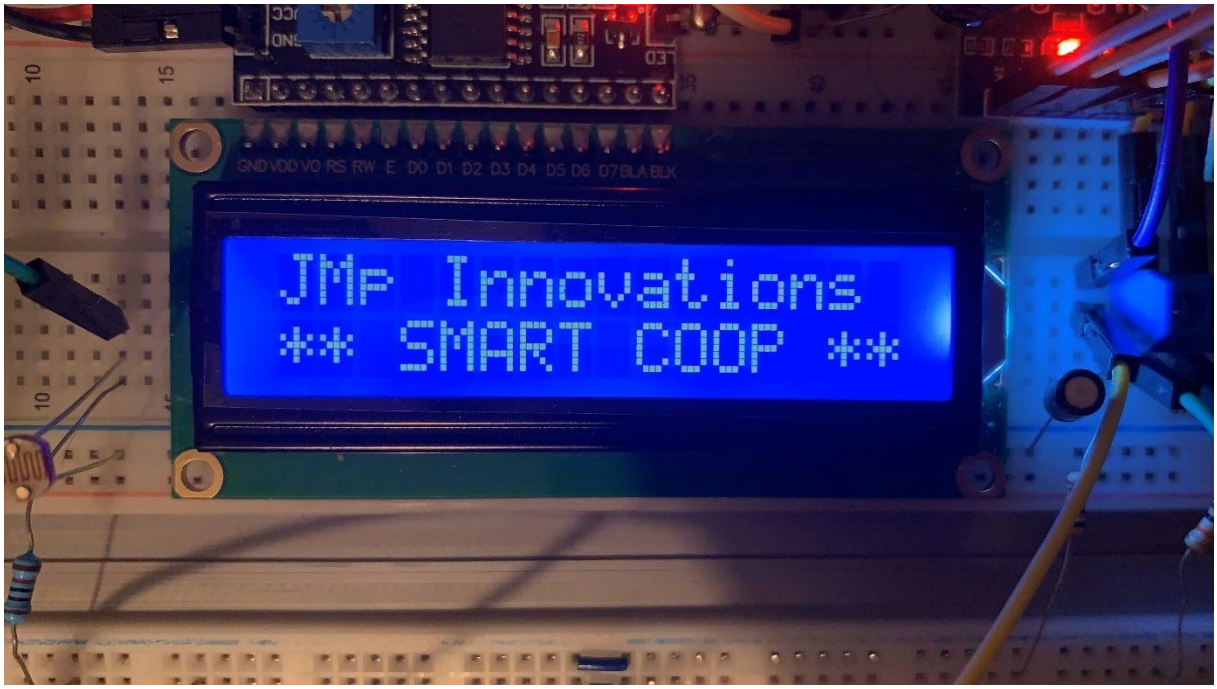
Η LCD 16x2 είναι ιδανική για την εμφάνιση απλών μηνυμάτων, αισθητηριακών δεδομένων, μενού, ενδείξεων σφαλμάτων ή κατάστασης συστήματος. Είναι ευανάγνωστη, σταθερή και φιλική προς τον χρήστη ακόμα και σε εξωτερικές ή ημιυπαίθριες εφαρμογές.

Ρόλος στην παρούσα εργασία:

Στην πτυχιακή σου εργασία, η LCD 16x2 χρησιμοποιείται για να εμφανίζει σε πραγματικό χρόνο κρίσιμα δεδομένα, όπως:

- Θερμοκρασία εντός του θαλάμου
- Ώρα/ημερομηνία
- Κατάσταση πόρτας (ανοιχτή/κλειστή)
- Πλήθος ορνίθων (μέσω RFID)
- Προειδοποιήσεις για νερό/τροφή

Αυτό παρέχει **άμεση οπτική πληροφόρηση** στον χρήστη επί τόπου, χωρίς να απαιτείται πρόσβαση στην εφαρμογή ή σε άλλες συνδεδεμένες συσκευές. Η ευκολία και αξιοπιστία της LCD την καθιστούν πολύτιμο εργαλείο για τον τελικό χειριστή



Εικόνα 39: Η lcd 16x2 στην παρούσα εργασία

3.3.1.2 LCD Display I2C Interface Module

Εικόνα 40: LCD I2C Interface Πηγή:https://www.grobotronics.gr

Η μονάδα **I2C interface για LCD** είναι ένα μικρό κύκλωμα που κουμπώνει επάνω σε μια συμβατική οθόνη LCD (όπως η LCD 16x2) και επιτρέπει τη σύνδεσή της με μικροελεγκτές μέσω του **πρωτοκόλλου I2C (Inter-Integrated Circuit)**. Το I2C είναι ένα πρωτόκολλο σειριακής επικοινωνίας που χρησιμοποιεί μόνο **δύο καλώδια**:

- **SDA (Serial Data)**
- **SCL (Serial Clock)**

Γενικά Χαρακτηριστικά:

- **Ελεγκτής**: Συνήθως βασίζεται στο chip **PCF8574** (I/O expander)
- **Τάση λειτουργίας**: 5V
- **Ρυθμιζόμενη αντίσταση (ποτενσιόμετρο)** για ρύθμιση της αντίθεσης της οθόνης
- **Διευθυνσιοδότηση I2C**: Δυνατότητα αλλαγής διεύθυνσης για χρήση πολλών συσκευών I2C ταυτόχρονα

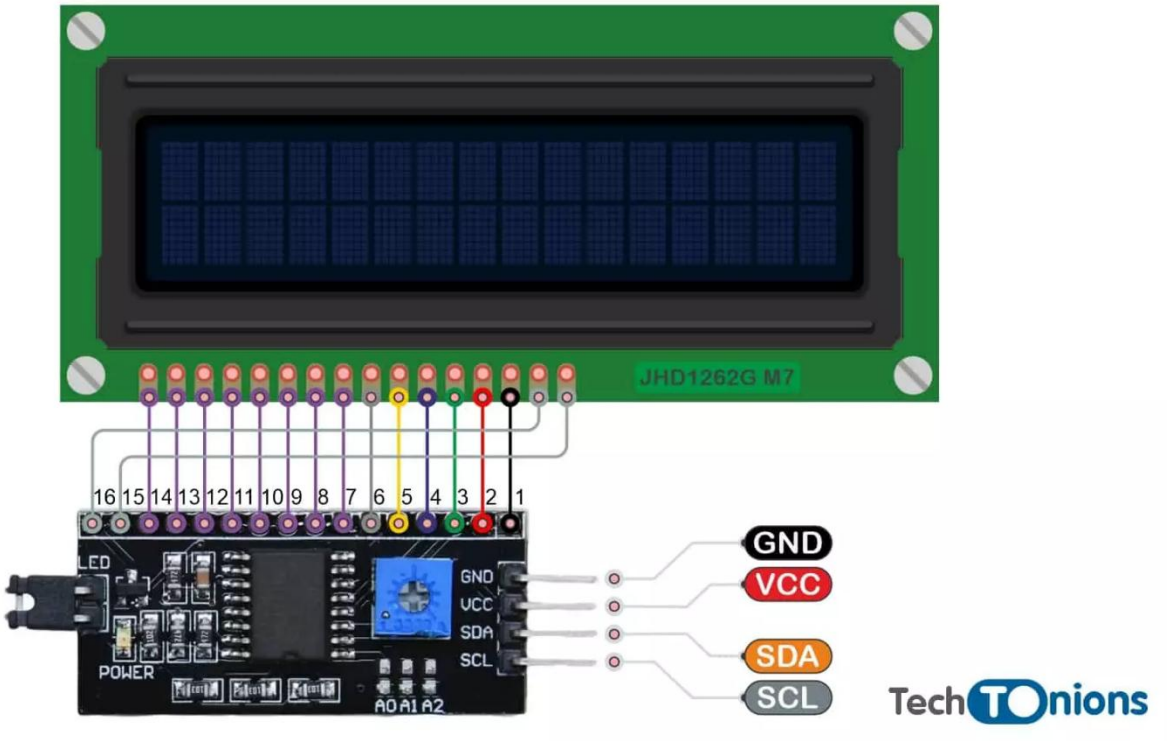
Πλεονεκτήματα:

- **Απλοποιεί σημαντικά τη συνδεσμολογία**: Αντί για 6 έως 10 καλώδια που απαιτούνται στην τυπική LCD, με το I2C χρειαζόμαστε μόνο 2.
- **Εξοικονόμηση θυρών (GPIOs)** στον μικροελεγκτή, κάτι ιδιαίτερα σημαντικό σε έργα με πολλαπλές συνδέσεις και αισθητήρες.
- **Ευκολότερος προγραμματισμός**, με έτοιμες βιβλιοθήκες για Arduino, ESP32 κ.ά. (π.χ. LiquidCrystal_I2C.h)

Χρήση στην πτυχιακή εργασία:

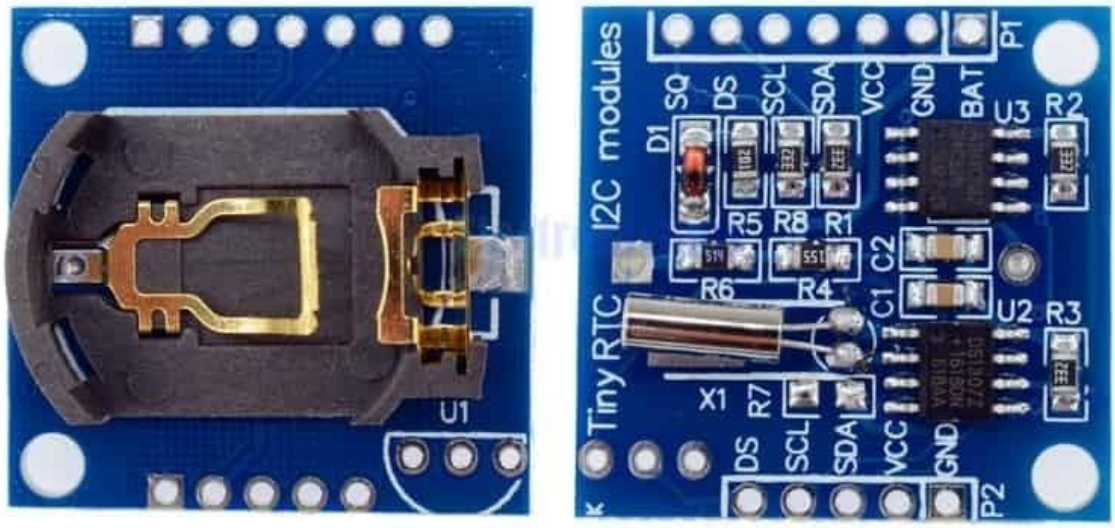
Στην παρούσα πτυχιακή, η μονάδα **I2C LCD Interface** επιτρέπει την εύκολη και λειτουργική σύνδεση της οθόνης LCD 16x2 με τον **μικροελεγκτή ESP32-C6**, καταλαμβάνοντας μόνο τις θύρες SDA και SCL. Έτσι, αφήνονται ελεύθερες περισσότερες θύρες για τους υπόλοιπους αισθητήρες και μηχανισμούς, κάτι κρίσιμο για τη συνολική ευστάθεια και επεκτασιμότητα του συστήματος.

Η μονάδα επίσης καθιστά ευκολότερη τη **συντήρηση και αποσφαλμάτωση** του κυκλώματος, αφού μειώνει την πολυπλοκότητα των καλωδιώσεων και αυξάνει την αξιοπιστία του hardware.



Εικόνα 41: LCD 16x2 & I2C Interface Wiring Πηγή: lcd 16x2 i2c connection - Αναζήτηση Google

3.3.1.3 Tiny RTC Module DS1307



Εικόνα 42: RTC1307 Πηγή:https://www.grobotronics.gr

Το **Tiny RTC Module DS1307** είναι ένα μικρό και οικονομικό κύκλωμα που λειτουργεί ως **ρολόι πραγματικού χρόνου (Real Time Clock - RTC)**. Η βασική του λειτουργία είναι να κρατάει με ακρίβεια την τρέχουσα ώρα, ημερομηνία, ημέρα της εβδομάδας, μήνα και έτος, ακόμη και όταν το κύριο σύστημα (μικροελεγκτής) είναι απενεργοποιημένο ή χωρίς τροφοδοσία.

Βασικά χαρακτηριστικά:

- **Ρολόι πραγματικού χρόνου** βασισμένο στο ολοκληρωμένο κύκλωμα **DS1307**.
- Επικοινωνία μέσω **πρωτοκόλλου I2C**, δηλαδή χρησιμοποιεί μόνο 2 γραμμές για σύνδεση (SDA, SCL).
- **Μπαταρία λιθίου (συνήθως CR2032)** για διατήρηση της ώρας ακόμα και αν το κύριο σύστημα δεν έχει ρεύμα.
- Υποστηρίζει λειτουργίες όπως 24ωρη/12ωρη μορφή, ημερομηνία, μήνα, έτος και ημέρα της εβδομάδας.
- Τάση λειτουργίας: **3.3V έως 5V**, κατάλληλο για χρήση με πολλούς μικροελεγκτές.

Γενική χρήση:

Το RTC είναι απαραίτητο σε συστήματα όπου η ακριβής παρακολούθηση χρόνου είναι κρίσιμη, ειδικά όταν η συσκευή μπορεί να απενεργοποιείται ή να επανεκκινείται συχνά. Χωρίς RTC, ο μικροελεγκτής θα χρειαζόταν σύνδεση με το διαδίκτυο για να συγχρονίσει την ώρα ή να ξεκινήσει από μηδενική βάση.

Παράδειγμα εφαρμογών περιλαμβάνουν:

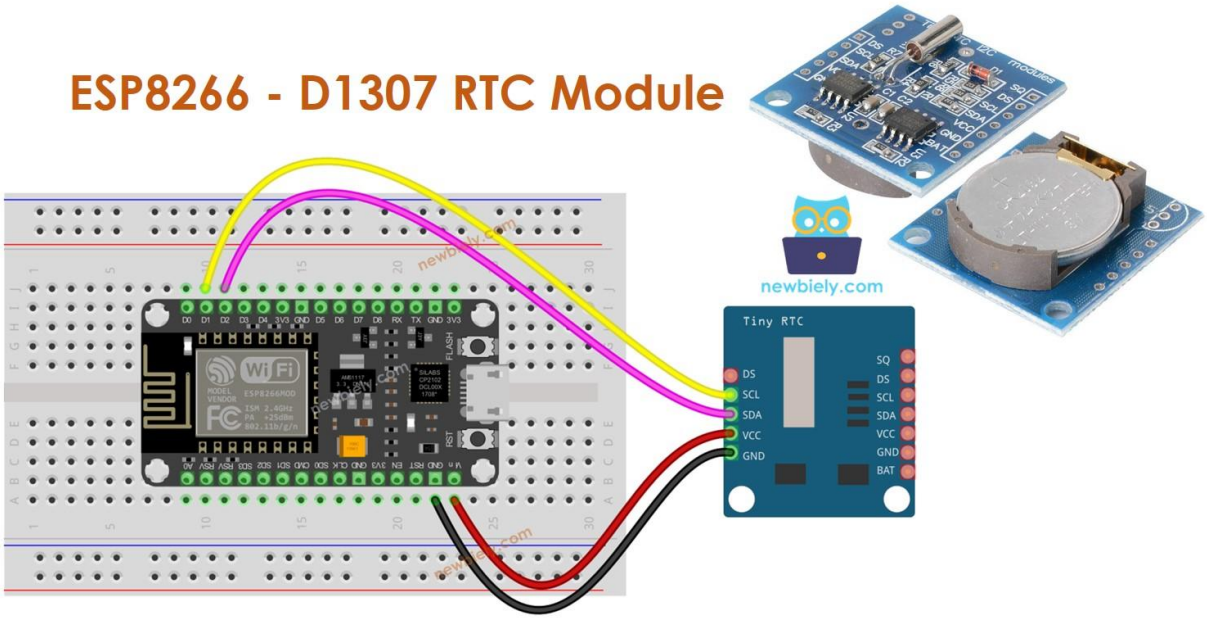
- Χρονοπρογραμματισμό λειτουργιών (π.χ. αυτόματη ενεργοποίηση/απενεργοποίηση)
- Καταγραφή δεδομένων με χρονική σήμανση
- Εφαρμογές όπου απαιτείται ιστορικό ή χρονοδιάγραμμα

Ρόλος στην πτυχιακή εργασία:

Στον αυτοματοποιημένο θάλαμο με όρνιθες, το **Tiny RTC Module DS1307** παρέχει την ακριβή χρονομέτρηση που απαιτείται για τον προγραμματισμό λειτουργιών όπως:

- Το άνοιγμα και κλείσιμο της πόρτας σε συγκεκριμένες ώρες.
- Το χρονοπρογραμματισμένο τάισμα των ορνίθων.
- Την εμφάνιση της τρέχουσας ώρας στην οθόνη LCD.

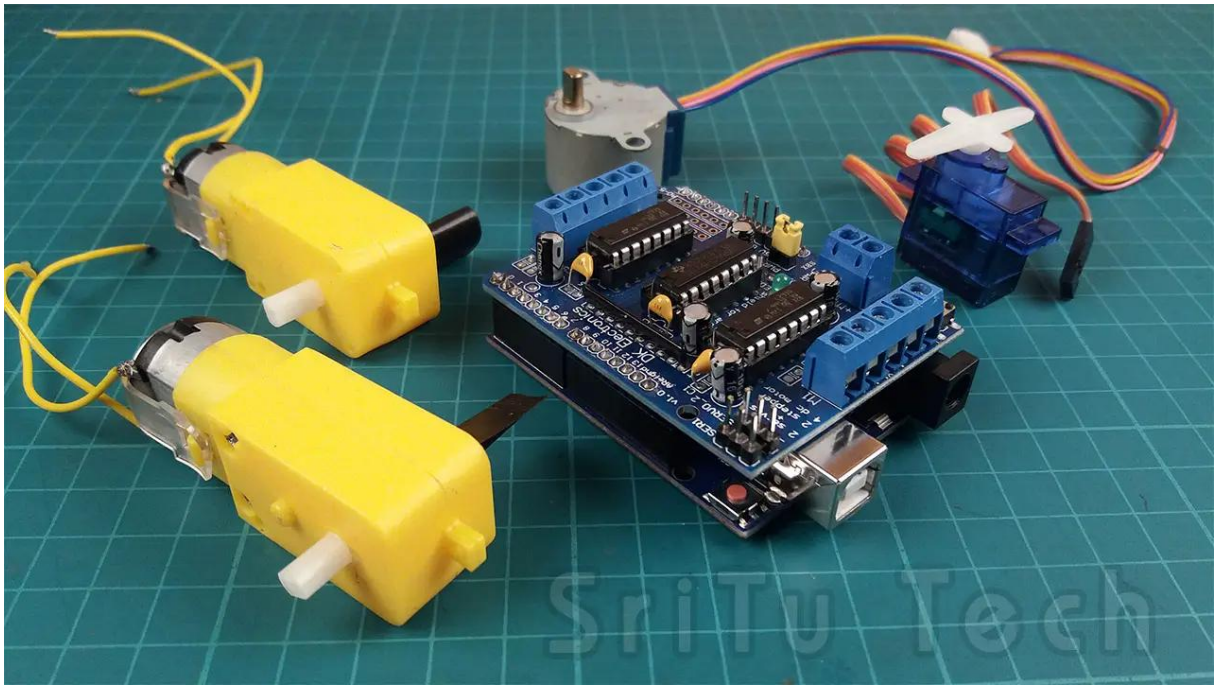
Χάρη στη μπαταρία του, το RTC συνεχίζει να κρατάει ακριβή χρόνο ακόμη και σε περιπτώσεις διακοπής ρεύματος, εξασφαλίζοντας έτσι την ομαλή λειτουργία και συνέχιση του προγράμματος του θαλάμου χωρίς ανάγκη επανασυγχρονισμού.



Εικόνα 43: ESP-RTC connection Πηγή: esp rtc ds1307 - Αναζήτηση Google

3.3.2 Κινητήρες και Οδηγοί Κίνησης

Οι κινητήρες και οι αντίστοιχοι οδηγοί τους (motor drivers) επιτρέπουν στο σύστημα να πραγματοποιεί φυσικές ενέργειες και μετακινήσεις, μετατρέποντας τα ηλεκτρικά σήματα σε μηχανική κίνηση. Στον αυτόματο θάλαμο, χρησιμοποιούνται κυρίως για την αυτοματοποιημένη κίνηση της πόρτας και άλλων στοιχείων που απαιτούν μηχανική παρέμβαση. Οι οδηγοί κινητήρων λειτουργούν ως ενδιάμεσοι μεταξύ μικροελεγκτή και κινητήρα, προσφέροντας την απαραίτητη ισχύ και προστασία. Μέσω αυτών, το σύστημα αποκτά τη δυνατότητα να **δρά**, πέρα από το να **αντιλαμβάνεται**, εξασφαλίζοντας αυτονομία και δυναμική απόκριση.



Εικόνα 44: Motor Driver With Motors Πηγή: motor and drivers arduino - Αναζήτηση Google

3.3.2.1 DC Gear Motor TT 120 RPM (μεταλλικός άξονας)

Εικόνα 45: TT Motor Duall Metal Shaft Πηγή:https://www.grobotronics.gr

Ο **DC Gear Motor TT** είναι ένας μικρός κινητήρας συνεχούς ρεύματος (DC motor) με ενσωματωμένο μηχανικό σύστημα μετάδοσης (γρανάζια), το οποίο μειώνει την ταχύτητα περιστροφής και αυξάνει τη ροπή (δυνατότητα δύναμης περιστροφής). Το συγκεκριμένο μοντέλο έχει **ταχύτητα 120 RPM (στροφές ανά λεπτό)** και μεταλλικούς άξονες διπλής εξόδου.

Βασικά χαρακτηριστικά:

- **Τάση λειτουργίας**: Συνήθως 3-6V, κατάλληλος για τροφοδοσία από μικρές μπαταρίες ή τροφοδοτικά χαμηλής τάσης.
- **Ταχύτητα περιστροφής**: 120 RPM, που σημαίνει ότι περιστρέφεται 120 φορές το λεπτό χωρίς φορτίο.
- **Μεταλλικοί άξονες διπλής εξόδου**: Δυνατότητα σύνδεσης τροχών ή άλλων μηχανισμών σε δύο άξονες, χρήσιμο για ρομποτικές εφαρμογές.
- **Ενσωματωμένο σύστημα γραναζιών**: Μειώνει την ταχύτητα και αυξάνει τη ροπή, επιτρέποντας στον κινητήρα να κινεί φορτία που απαιτούν περισσότερη δύναμη.

Γενική χρήση:

Αυτός ο τύπος κινητήρα χρησιμοποιείται ευρέως σε εκπαιδευτικά και ερασιτεχνικά ρομποτικά έργα, σε μικρές μηχανικές κατασκευές και αυτοματισμούς όπου χρειάζεται ακριβής και ελεγχόμενη κίνηση με επαρκή ροπή.

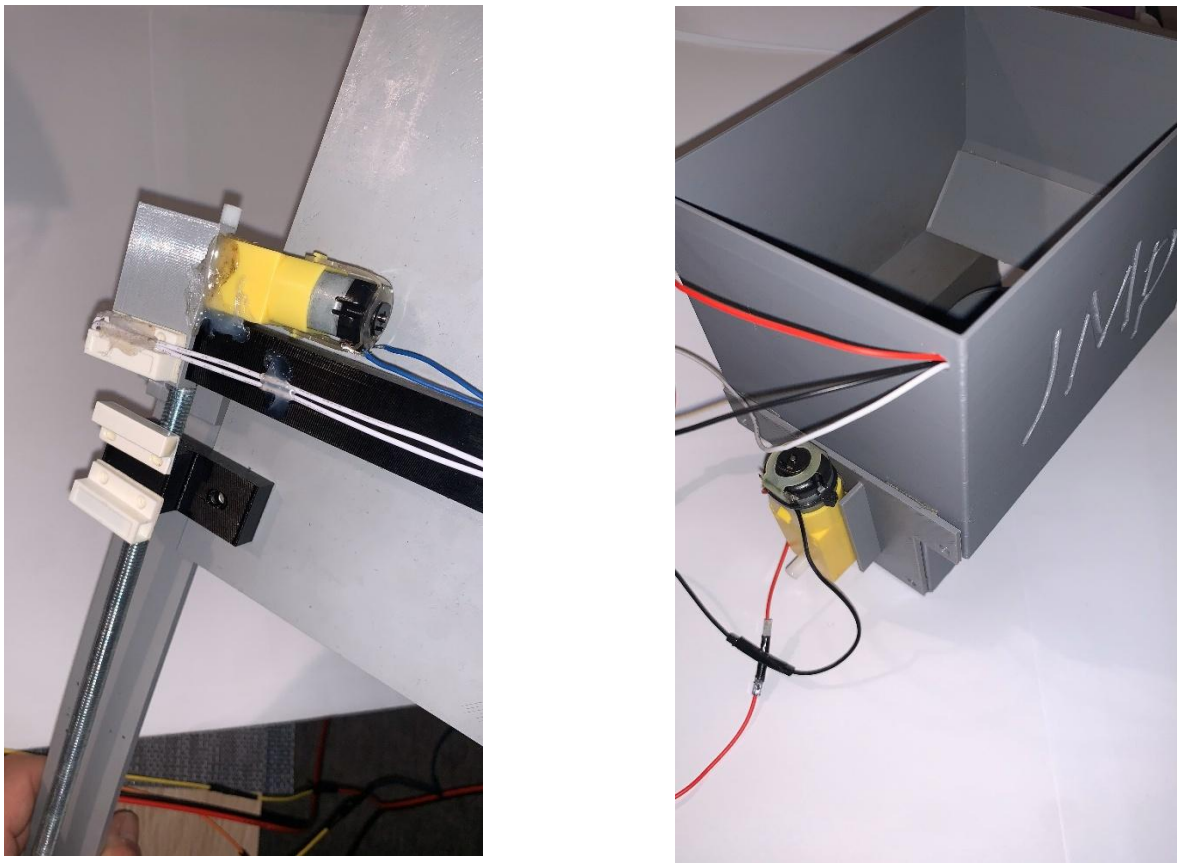
Οι κινητήρες με γρανάζια, όπως ο TT, είναι ιδανικοί για εφαρμογές όπου η ταχύτητα πρέπει να μειωθεί και η δύναμη να αυξηθεί σε σχέση με έναν απλό κινητήρα DC, όπως σε ρομποτικά οχήματα, αυτόματες πόρτες ή μηχανισμούς κίνησης.

Ρόλος στην πτυχιακή εργασία:

Στον αυτοματοποιημένο θάλαμο με όρνιθες, ο **DC Gear Motor TT** είναι ο κινητήρας που κινεί την πόρτα του θαλάμου. Λόγω της ενσωματωμένης μετάδοσης, μπορεί να:

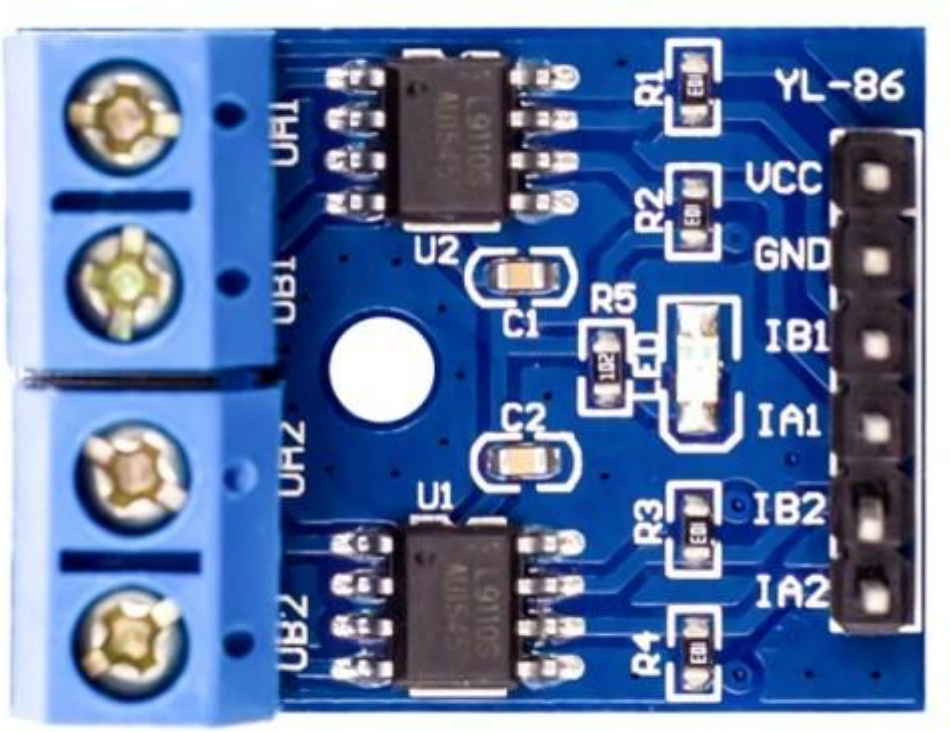
- Ανοίγει και κλείνει την πόρτα με τη σωστή ταχύτητα και δύναμη, ώστε να μην προκληθούν ζημιές.
- Αντιμετωπίζει την αντίσταση από την πόρτα και τυχόν καιρικές συνθήκες, όπως αέρας ή μικρά εμπόδια.
- Λειτουργεί αποδοτικά σε χαμηλή τάση, συμβάλλοντας στη συνολική ενεργειακή αυτονομία του συστήματος.

Η σύνδεση με το **Dual Motor Driver Module L9110S** επιτρέπει τον ακριβή έλεγχο της φοράς και της λειτουργίας του κινητήρα, καθιστώντας το σύστημα αξιόπιστο και ευέλικτο.



Εικόνα 46: Οι Δυο DC TT MOTOR που χρησιμοποιήθηκαν στην εργασία

3.3.2.2 Dual Motor Driver L9110S (οδήγηση κινητήρων)



Εικόνα 47: L9110S Motor Driver Module Πηγή: https://www.grobotronics.gr

Το **Dual Motor Driver Module L9110S** είναι ένα μικρό και αποδοτικό κύκλωμα ελέγχου κινητήρων συνεχούς ρεύματος (DC motors). Είναι σχεδιασμένο για να επιτρέπει τον έλεγχο της φοράς και της ταχύτητας δύο κινητήρων ανεξάρτητα, χρησιμοποιώντας έναν μικροελεγκτή.

Βασικά χαρακτηριστικά:

- Μπορεί να οδηγήσει **δύο κινητήρες DC** ταυτόχρονα.
- Υποστηρίζει λειτουργία **εμπρός και πίσω** (ανάποδη φορά) για κάθε κινητήρα.
- Τάση λειτουργίας: συνήθως **2.5V έως 12V**, καθιστώντας το κατάλληλο για μικρούς κινητήρες όπως ο DC Gear Motor TT που χρησιμοποιείται στην εργασία.
- Μέγιστο ρεύμα εξόδου ανά κανάλι περίπου **800mA**, αρκετό για μικρούς κινητήρες.
- Πολύ χαμηλή κατανάλωση ενέργειας και θερμότητα κατά τη λειτουργία.

Γενική χρήση:

Ο driver αυτός χρησιμοποιείται σε έργα ρομποτικής και αυτοματισμού όπου απαιτείται ο έλεγχος της κίνησης κινητήρων DC. Παρέχει τα απαραίτητα σήματα και προστασία ώστε να ελέγχονται με ασφάλεια και ακρίβεια οι κινητήρες από μικροελεγκτές που δεν μπορούν να τους τροφοδοτήσουν απευθείας λόγω περιορισμένων ρευμάτων.

Μπορεί να ελέγχει:

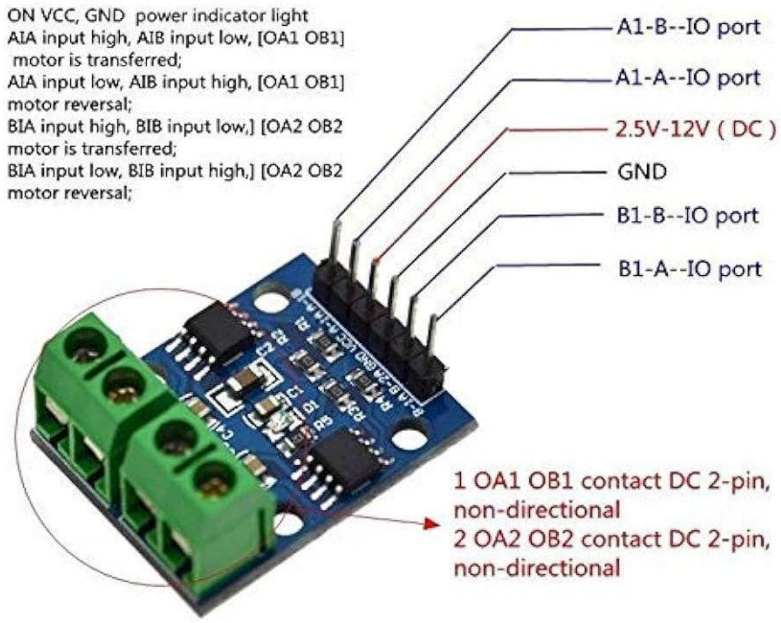
- Την κατεύθυνση περιστροφής (δεξιά ή αριστερά).
- Την ενεργοποίηση ή απενεργοποίηση του κινητήρα.
- Μέσω PWM (Pulse Width Modulation), την ταχύτητα περιστροφής.

Ρόλος στην πτυχιακή εργασία:

Στον αυτοματοποιημένο θάλαμο με όρνιθες, το **Dual Motor Driver L9110S** είναι υπεύθυνο για την κίνηση της **αυτόματης πόρτας** και της ταΐστρας μέσω του κινητήρα DC Gear Motor TT. Ο μικροελεγκτής δίνει εντολές στον driver ώστε να:

- Ανοίγει ή κλείνει την πόρτα ανάλογα με τις προγραμματισμένες ώρες ή τις εντολές από τον χρήστη.
- Αλλάζει κατεύθυνση περιστροφής για το άνοιγμα και το κλείσιμο.
- Διασφαλίζει ομαλή και ελεγχόμενη κίνηση, αποφεύγοντας φθορές ή απρόβλεπτες καταστάσεις.

Η χρήση του L9110S διευκολύνει την ολοκλήρωση του συστήματος, αφού ο μικροελεγκτής δεν χρειάζεται να χειριστεί άμεσα το μεγάλο ρεύμα του κινητήρα, εξασφαλίζοντας έτσι μεγαλύτερη αξιοπιστία και προστασία των ηλεκτρονικών εξαρτημάτων.



Εικόνα 48: L9110S Pin Out Πηγή: l9110s dc motor drive module - Αναζήτηση Google

3.3.3 Συστήματα Εισόδου και Χρήστη

Τα συστήματα εισόδου αποτελούν τον κύριο τρόπο με τον οποίο ο χρήστης μπορεί να επικοινωνήσει άμεσα με το σύστημα. Περιλαμβάνουν διακόπτες (buttons), μαγνητικούς αισθητήρες ή άλλες μορφές εισόδου που ενεργοποιούν χειροκίνητες λειτουργίες, δίνουν εντολές ή επιτρέπουν τη ρύθμιση παραμέτρων. Αυτά τα μέσα είναι ιδιαίτερα σημαντικά σε περιβάλλοντα όπου απαιτείται βασικός έλεγχος χωρίς τη χρήση εφαρμογών ή δικτύου, προσφέροντας ευκολία και αξιοπιστία. Στην παρούσα εργασία, τέτοια συστήματα λειτουργούν υποστηρικτικά, δίνοντας στον χρήστη άμεση αλληλεπίδραση και έλεγχο όταν χρειαστεί.

3.3.3.1 Push Buttons/Διακόπτες (χειροκίνητη παρέμβαση, λειτουργίες)

Εικόνα 49: Tact Switch Πηγή: https://grobotronics.com/tact-switch-6x6mm-5mmbtt-a06-5.0.html

Τα **κουμπιά** (ή πλήκτρα, στα αγγλικά *push buttons*) είναι απλά ηλεκτρομηχανικά εξαρτήματα που λειτουργούν ως **διακόπτες στιγμιαίας επαφής**. Όταν πατηθούν, **κλείνουν το κύκλωμα**, επιτρέποντας τη ροή του ρεύματος. Όταν απελευθερωθούν, το κύκλωμα **ανοίγει** ξανά.

Γενικά χαρακτηριστικά:

- **Τύπος επαφής**: στιγμιαίας λειτουργίας (momentary switch).
- **Τετράποδα ή δίποδα**.
- **Μικρού μεγέθους**, εύκολα ενσωματώσιμα σε breadboard ή PCB.
- Συχνά χρειάζονται **αντιστάσεις pull-up ή pull-down** για σωστή ανίχνευση σήματος.

Γενική χρήση:

- **Έναρξη/παύση λειτουργιών** σε μικροελεγκτές ή συσκευές.
- **Ανθρώπινη αλληλεπίδραση** με το σύστημα (HMI - Human Machine Interface).
- **Καθορισμός καταστάσεων** (π.χ. χειροκίνητη/αυτόματη λειτουργία).
- **Επαναφορά (reset)** ή αλλαγή παραμέτρων.

Χρήση στην πτυχιακή εργασία:

Τα κουμπιά χρησιμοποιούνται για:

- **Χειροκίνητη ενεργοποίηση λειτουργιών**, όπως:
    - Άνοιγμα/κλείσιμο της πόρτας του θαλάμου.
    - Χειροκίνητη εκκίνηση ταΐσματος.
- **Εναλλαγή μεταξύ αυτόματης και χειροκίνητης λειτουργίας**.
- **Προγραμματισμός/διαμόρφωση ρυθμίσεων**, όπως:
    - Σύνδεση Wi-Fi.
    - Καταγραφή αριθμού ορνίθων.
    - Ρυθμίσεις προστασίας από καιρικές συνθήκες.

Τα κουμπιά επικοινωνούν απευθείας με τον ESP32, και κάθε πίεση **ανιχνεύεται με κατάλληλο προγραμματισμό** που επιτρέπει στο σύστημα να ανταποκρίνεται άμεσα ή να αλλάζει λειτουργία.

Τεχνική σημείωση:

Στον προγραμματισμό συστημάτων με κουμπιά, λαμβάνεται υπόψη και το φαινόμενο **"debouncing"** – δηλαδή, η **καθαρή ανάγνωση** της πίεσης χωρίς ανεπιθύμητες επαναλαμβανόμενες εντολές, κάτι που επιτυγχάνεται είτε με **καθυστέρηση στον κώδικα (software)** είτε με **πυκνωτές εξομάλυνσης (hardware)**.

3.4 Παθητικά Ηλεκτρονικά Στοιχεία

Τα παθητικά εξαρτήματα, όπως **αντιστάσεις** και **πυκνωτές**, μπορεί να μην είναι εμφανώς «έξυπνα», αλλά διαδραματίζουν κρίσιμο ρόλο στη σταθερότητα και λειτουργικότητα κάθε ηλεκτρονικού κυκλώματος. Οι αντιστάσεις χρησιμοποιούνται για τον έλεγχο της ροής ρεύματος και την προστασία ευαίσθητων εξαρτημάτων, ενώ οι πυκνωτές συμβάλλουν στην εξομάλυνση της τάσης, τη μείωση του ηλεκτρικού θορύβου και τη σταθερότητα των παλμών.
Χωρίς τη σωστή χρήση παθητικών στοιχείων, το σύστημα μπορεί να παρουσιάσει ασταθή συμπεριφορά ή ακόμη και να υποστεί φθορά με την πάροδο του χρόνου. Για τον λόγο αυτό, τα παθητικά στοιχεία είναι απαραίτητα για την αξιόπιστη και μακροχρόνια λειτουργία κάθε μικροελεγκτικού συστήματος, ειδικά όταν αυτό λειτουργεί σε εξωτερικές ή μεταβαλλόμενες συνθήκες όπως αυτές ενός θαλάμου με όρνιθες..

Εικόνα 50: Passive Components Πηγή: passive components - Αναζήτηση Google

3.4.1 Αντιστάσεις (Resistors)

Εικόνα 51: Resistor Πηγή:https://www.grobotronics.gr

Οι **αντιστάσεις** είναι από τα πιο βασικά και ευρέως χρησιμοποιούμενα εξαρτήματα στην ηλεκτρονική. Ο ρόλος τους είναι να **περιορίζουν τη ροή του ηλεκτρικού ρεύματος** σε ένα κύκλωμα και να **προκαλούν πτώση τάσης**, όταν χρειάζεται.

Γενικά χαρακτηριστικά:

- Εκφράζονται σε **Ohm (Ω)**.
- Υπάρχουν σε διάφορες ισχύς (π.χ. 1/4W, 1/2W).
- Υπάρχουν σταθερές και μεταβλητές (ποτενσιόμετρα).
- Αναγνωρίζονται από **έγχρωμους δακτυλίους** που υποδεικνύουν την τιμή τους.

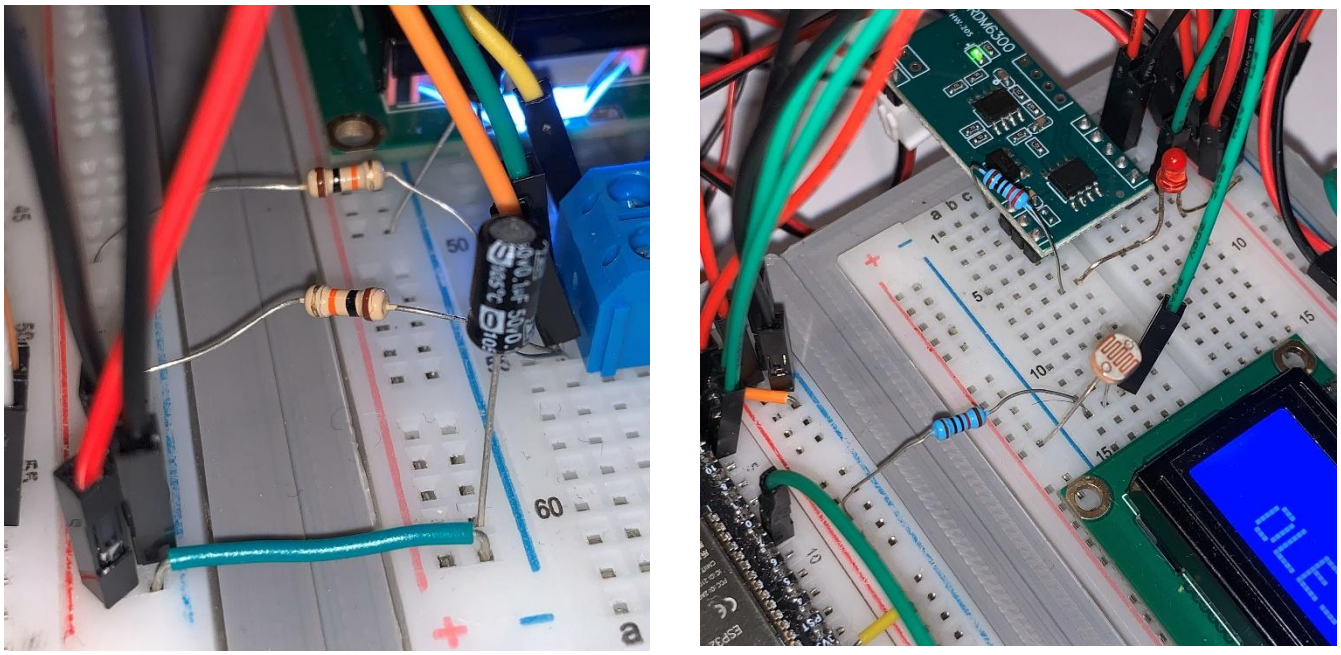
Γενική χρήση:

- Περιορισμός ρεύματος σε LED (ώστε να μη καούν).
- Δημιουργία διαιρέσεων τάσης.
- Απόσβεση σημάτων ή φίλτρα.
- Προστασία κυκλωμάτων από υπερφόρτωση.

Χρήση στην πτυχιακή εργασία:

Στην παρούσα κατασκευή, οι αντιστάσεις χρησιμοποιούνται:

- Στη **σύνδεση φωτοαντίστασης (LDR)**, για τη δημιουργία **διαιρέτη τάσης** ώστε να "διαβάζεται" η φωτεινότητα.
- Στους **αισθητήρες IR** για σωστή ρύθμιση του ρεύματος προς τα LED.
- Στην **προστασία εισόδων** του μικροελεγκτή από υπερένταση.
- Στη **ρύθμιση στάθμης σημάτων**, ώστε να είναι συμβατά με τα λογικά επίπεδα του ESP32.



Εικόνα 52: Οι Αντιστάσεις Που Χρησιμοποιήθηκαν Στην Εργασία

3.4.2 Πυκνωτές (Capacitors)

Εικόνα 53: capacitor Πηγή:https://www.grobotronics.gr

Οι **πυκνωτές** είναι παθητικά εξαρτήματα που **αποθηκεύουν ηλεκτρική ενέργεια** σε μορφή ηλεκτρικού πεδίου. Χρησιμοποιούνται ευρέως για **φιλτράρισμα, σταθεροποίηση τάσης, αποθορυβοποίηση και καθυστέρηση σημάτων**.

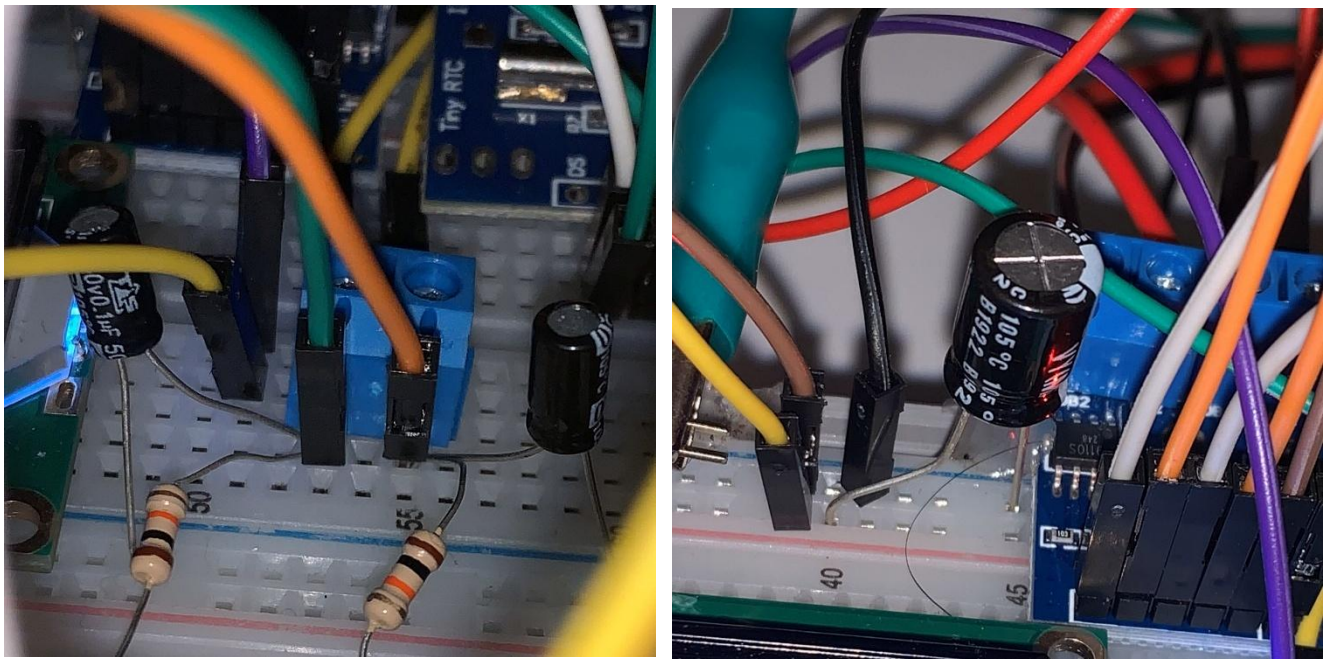
Γενικά χαρακτηριστικά:

- Εκφράζονται σε **Farads (F)**, πιο συχνά σε **μικρο** ή **νανο**.
- Υπάρχουν **κεραμικοί, ηλεκτρολυτικοί, πολυεστερικοί**, κ.ά.
- Οι ηλεκτρολυτικοί έχουν **πολικότητα** (θετικός-αρνητικός ακροδέκτης).
- **Γενική χρήση:**
- **Φιλτράρισμα** και **ομαλοποίηση** σε τροφοδοτικά (π.χ. εξομάλυνση μεταβλητής τάσης).
- **Παράκαμψη (bypass)** θορύβων από ευαίσθητα κυκλώματα.
- **Αποθήκευση ενέργειας** για στιγμιαία παροχή ισχύος.
- **Χρονισμός** σε συνδυασμό με αντιστάσεις (RC κύκλωμα).

Χρήση στην πτυχιακή εργασία:

Οι πυκνωτές χρησιμοποιούνται:

- Στα **κυκλώματα τροφοδοσίας** (μαζί με το φωτοβολταϊκό και μπαταρίες) για **φιλτράρισμα και σταθεροποίηση** της τάσης.
- Σε **ευαίσθητες εισόδους** του μικροελεγκτή (π.χ. από αισθητήρες) για **αποθορυβοποίηση**.



Εικόνα 54: Οι Πυκνωτές (ηλεκτρολυτικοί) Που Χρησιμοποιήθηκαν Στην Εργασία

3.5 Επίλογος Κεφαλαίου

Το κεφάλαιο αυτό ανέδειξε τη σημαντική θέση που κατέχουν οι αισθητήρες και τα περιφερειακά σε ένα αυτοματοποιημένο σύστημα, όπως ο έξυπνος θάλαμος ορνίθων. Οι αισθητήρες αποτελούν τα «μάτια» και «αυτιά» του συστήματος, συλλέγοντας πολύτιμα δεδομένα από το περιβάλλον, ενώ τα περιφερειακά και ενεργά μέρη δίνουν τη δυνατότητα εκτέλεσης ενεργειών και αλληλεπίδρασης με τον χρήστη. Η σωστή επιλογή και ενσωμάτωση κάθε εξαρτήματος εξασφαλίζει τη λειτουργικότητα, την αξιοπιστία και την ευχρηστία της συνολικής εφαρμογής. Με την ολοκλήρωση αυτής της θεμελιώδους βάσης, το επόμενο βήμα είναι η υλοποίηση και η διασύνδεση των συστημάτων, που θα εξεταστεί αναλυτικά στο επόμενο κεφάλαιο.

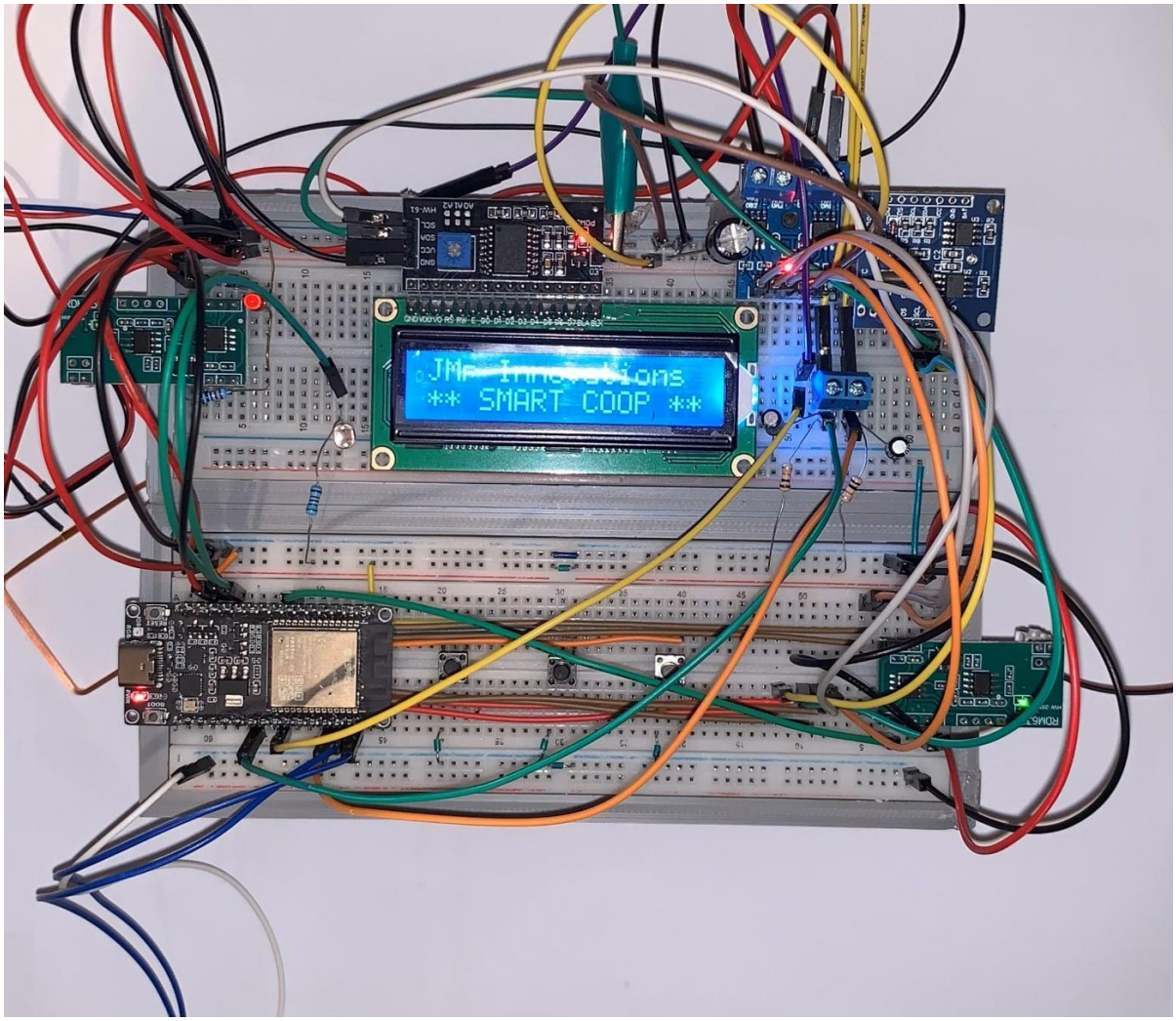
Κεφάλαιο 4ο: Σχεδιασμός και υλοποίηση θαλάμου

Σε αυτό το κεφάλαιο θα μιλήσουμε για την κατασκευή του θαλάμου, τα δύο βασικά επιμέρους τμήματα της, το ηλεκτρονικό κομμάτι και το κατασκευαστικό κομμάτι καθώς και την συναρμολόγηση του. Η αρχική ιδέα είναι να γίνει ένας θάλαμος οποίος θα έχει μεγάλη ευκολία στην μεταφορά του και στην συναρμολόγηση του. Κατασκευάστηκε λοιπόν ένας θάλαμος ο οποίος είναι χωρισμένος σε τμήματα συναρμολογείτε μέσα σε μικρό χρονικό διάστημα χωρίς ιδιαίτερη γνώση σε κατασκευές και χωρίς την ανάγκη για χρήση εργαλείων. Η χωρητικότητα του είναι για τέσσερις με πέντε κότες και είναι αυτόνομο με χρήση φωτοβολταϊκού συστήματος και μπαταρίας. Επιμέρους κομμάτια όπως η ταΐστρα και η πόρτα έχουν κατασκευαστεί εξολοκλήρου σε 3-D εκτυπωτή έχουν σχεδιαστεί από εμένα με ιδέες που έχω δει κατά καιρούς στο διαδίκτυο Και έχουν συνδεθεί παντρεύοντας το μηχανολογικό κομμάτι με τις γνώσεις του ηλεκτρονικού μηχανικού δημιουργώντας έτσι ένα Smart σύστημα αυτοματοποιημένο θαλάμου για όρνιθες. Το συνολικό σύστημα αποτελείται από πέντε πλαίσια διαστάσεων ένα επί ένα, τέσσερα πόδια Μήκους ενάμιση μέτρου ,την οροφή, την ταΐστρα, την πόρτα, την ποτίστρα και το κεντρικό σύστημα ελέγχου. Στην οροφή υπάρχει εγκατεστημένο φωτοβολταϊκό Πάνελ απόδοσης 30 W συνδεδεμένο με ρυθμιστή φόρτισης και μπαταρία 12 V. Το σύστημα τροφοδοτείται από την έξοδο 5 V που έχει ενσωματωμένη ο ρυθμιστής φόρτισης. Ο θάλαμος είναι υπερυψωμένος κατά μισό μέτρο προστατεύοντας έτσι τις κότες από επιθέσεις ζώων από χαμηλά όπως τρωκτικά, στην είσοδο του έχει δύο κεραίες που είναι συνδεδεμένες με το σύστημα RFID reader Προκειμένου να ελέγχουν και κότες είναι μέσα και ποιες είναι έξω. Το σύστημα περιλαμβάνει επίσης αισθητήρα θερμοκρασίας το οποίο ανιχνεύει θερμοκρασία χαμηλότερη του μηδενός μέσω θερμαντικού σώματος ζεσταίνει το νερό προκειμένου αυτό να μην παγώσει. Έχει επίσης λειτουργία προστασίας από δυσμενείς καιρικές συνθήκες όπως σε περίπτωση έντονης βροχής χιονιού η χαμηλής θερμοκρασίας κρατάει την πόρτα κλειστή προκειμένου να μείνουν προστατευμένες οι κότες

Εικόνα 55: Μπροστά όψη Του Θαλάμου

4.1 Κατασκευή Κεντρικής μονάδας και συνδεσμολογία Ηλεκτρονικών εξαρτημάτων

Σε αυτή την ενότητα παρουσιάζεται η κατασκευή της "καρδιάς" του συστήματος — της ηλεκτρονικής μονάδας ελέγχου. Η κεντρική πλακέτα με τον μικροελεγκτή ESP32C6 DevKit N8 αποτελεί το βασικό σημείο διαχείρισης και επικοινωνίας όλων των περιφερειακών συσκευών και αισθητήρων. Η ολοκλήρωση αυτής της ενότητας προσφέρει ένα πλήρως λειτουργικό "νευρικό σύστημα" έτοιμο να εφαρμοστεί στο φυσικό περιβάλλον του θαλάμου.



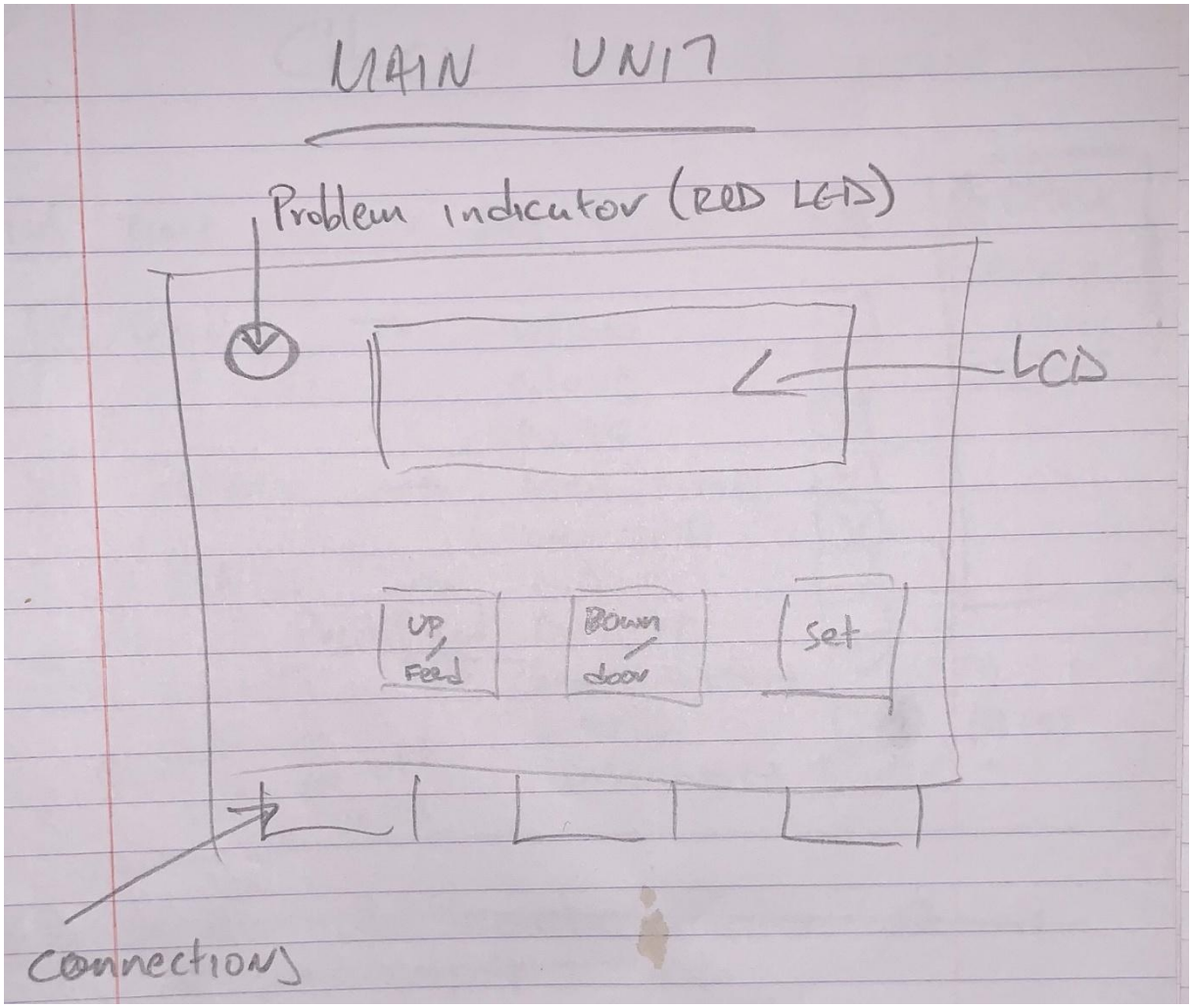
Εικόνα 56: Το Κύκλωμα Της Κεντρικής Μονάδας Στην Πλακέτα Δοκιμών

4.1.1 Κεντρική Μονάδα

Η κεντρική μονάδα αποτελεί τον πυρήνα του αυτοματοποιημένου συστήματος και φιλοξενεί τον μικροελεγκτή ESP32-C6 DevKit N8, ο οποίος είναι υπεύθυνος για τη λήψη, επεξεργασία και μετάδοση των δεδομένων που προέρχονται από αισθητήρες και περιφερειακά. Η επιλογή του ESP32 έγινε λόγω των ισχυρών χαρακτηριστικών του (Wi-Fi, Bluetooth, επαρκείς είσοδοι/έξοδοι και χαμηλή κατανάλωση ενέργειας), που τον καθιστούν ιδανικό για εφαρμογές IoT και αυτοματισμού.

Η κατασκευή της κεντρικής μονάδας έγινε αρχικά σε πλακέτα δοκιμών χρησιμοποιώντας δύο από αυτές για να μπορέσουν να συνδεθούν όλα τα υλικά πάνω μαζί με το esp32. Η ιδέα ήταν να χρησιμοποιηθούν τρία κουμπιά το up, το down και το select.

Πατώντας συνεχόμενα το select μπαίνουμε στο μενού ρυθμίσεων και από κει με τα κουμπιά πάνω και κάτω μπορούμε να ρυθμίζουμε την ώρα, πότε θα ανοίξει η πόρτα, πότε θα κλείσει, αν θα είναι αυτόματη, πότε θα ταΐσει η ταΐστρα, αν θα Είναι ενεργή η ταΐστρα, την προστασία από καιρικές συνθήκες τον αριθμό στις κότες που έχουμε μέσα στο κοτέτσι και την σύνδεση με το wifi. Επίσης έχουμε τη δυνατότητα για manual feeding πατώντας συνεχόμενα το κουμπί πάνω ή μπορούμε να ανοίξουμε/κλείσουμε την πόρτα πατώντας συνεχόμενα το κουμπί κάτω. Η κεντρική μονάδα είναι επίσης υπεύθυνη για να λαμβάνει όλα τα επιμέρους δεδομένα από τους ανάλογους αισθητήρες, να τα επεξεργάζεται και να παίρνει τις κατάλληλες αποφάσεις για εμάς.



Εικόνα 57: Πρόχειρο σχέδιο τελικής μορφής

Στην συνέχεια κεντρική μονάδα με όλα τα παρελκόμενα της μεταφέρθηκε σε διάτρητη πλακέτα και εκεί φτιάχτηκε η τελική της μορφή όπου με την βοήθεια 3-D εκτυπωτή κατασκευάστηκε το Περίβλημα της προκειμένου να τοποθετηθεί με ασφάλεια στο θάλαμο

4.1.2 Συνδεσμολογία ηλεκτρονικών

Η συνδεσμολογία των ηλεκτρονικών εξαρτημάτων αποτελεί βασικό στάδιο στην κατασκευή του αυτόματου θαλάμου, καθώς διαμορφώνει το πρακτικό δίκτυο επικοινωνίας ανάμεσα στον μικροελεγκτή και όλα τα συνδεδεμένα υποσυστήματα. Η διαδικασία περιλάμβανε τη φυσική διασύνδεση των αισθητήρων, των περιφερειακών και των ενεργών στοιχείων με την κεντρική μονάδα, έτσι ώστε να μπορούν να συνεργάζονται αποδοτικά και με συνέπεια. Για την πλοήγηση στη συνδεσμολογία αξιοποιήθηκαν βασικά διαγράμματα που βοήθησαν στην οργάνωση των εισόδων και εξόδων του ESP32, ενώ η καλωδίωση έγινε με βάση τις λειτουργικές ανάγκες κάθε υπομονάδας (αισθητήρες, οθόνες, κινητήρες, κουμπιά κ.ά.).

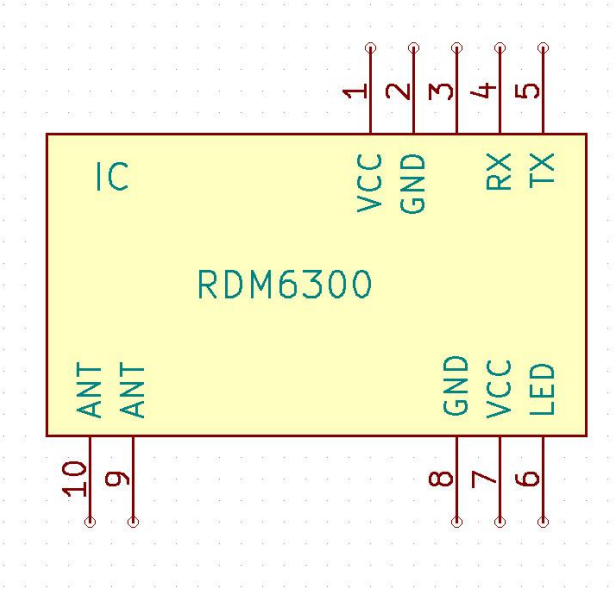
Στον παρακάτω πίνακα ακολουθούν οι συνδεσμολογίες του κυκλώματος με τον μικροελεκτη

ΣΥΝΔΕΣΜΟΛΟΓΙΑ GPIO PINS ΜΕ ΤΑ ΑΙΣΘΗΤΗΡΙΑ	
ESP GPIO	COMPOMENT
0	TX RD6300 (ΕΙΣΟΔΟΣ)
1	THERMISTOR
2	TX RD6300 (ΕΞΟΔΟΣ)
3	LDR
4	BUTTON UP (INTERNAL PULL-UP)
5	BUTTON DOWN (INTERNAL PULL-UP)
6	BUTTON SELECT(INTERNAL PULL-UP)
7	WATER LEVEL SENSOR (INTERNAL PULL-UP)
8	IB2 MOTOR DRIVER
9	SYSTEM BOOT
10	IA2 MOTOR DRIVER
11	RELAY INPUT
12	IB1 MOTOR DRIVER
13	IA1 MOTOR DRIVER
14	EMPTY
15	DOOR MAGNET TOP POSITION(INT PULL-UP)
16	SYSTEM TX
17	SYSTEM RX
18	DOOR MAGNET DOWN POSITION (INT PULL-UP)
19	WARNING LED
20	IR BREAK BEAM
21	I2C SDA
22	I2C SCL
23	DIGITAL OUT RAIN SENSOR

Πίνακας 1: Συνδεσεις GPIO με Compoments

Και εδώ ακολουθούν οι επιμέρους συνδέσεις

RDM6300 RFID READER

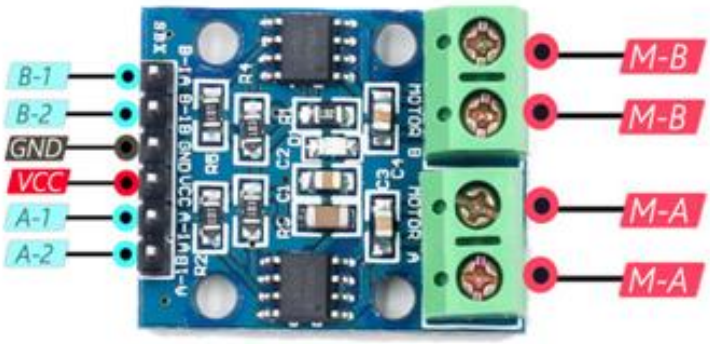


Εικόνα 58: RDM6300 .

RDM6300		
PIN	SYMBOL	CONN
1	VCC	+3.3V
2	GND	GND
3		KENO
4	RX	KENO
5	TX	GPIO 0 & GPIO2
6	LED	KENO
7	VCC	KENO
8	GND	KENO
9	ANT	KEPAIA
10	ANT	KEPAIA

Πίνακας 2: Συνδέσεις RDM6300

L9110S MOTOR DRIVER

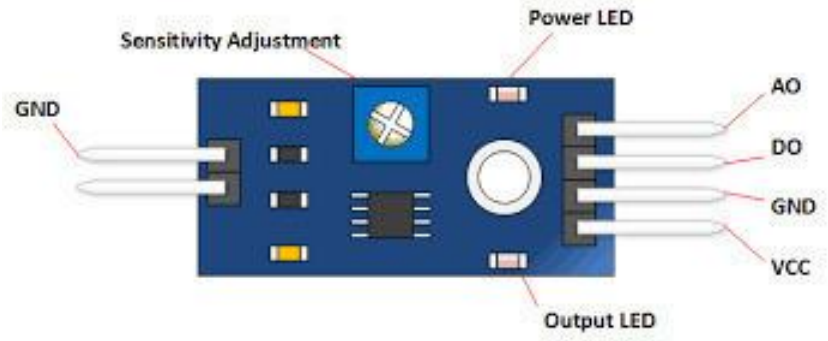


Εικόνα 59: L9110S Pin Out

L9110S		
PIN	SYMBOL	CONN
1	B-1	GPIO8
2	B-2	GPIO10
3	GND	GND
4	VCC	5V
5	A-1	GPIO12
6	A-2	GPIO13
7	M-B	DOOR MOTOR
8	M-B	DOOR MOTOR
9	M-A	FEEDER MOTOR
10	M-A	FEEDER MOTOR

Πίνακας 3: Συνδέσεις με Motor Driver L9110s

MH-RD RAIN & SNOW SENSOR

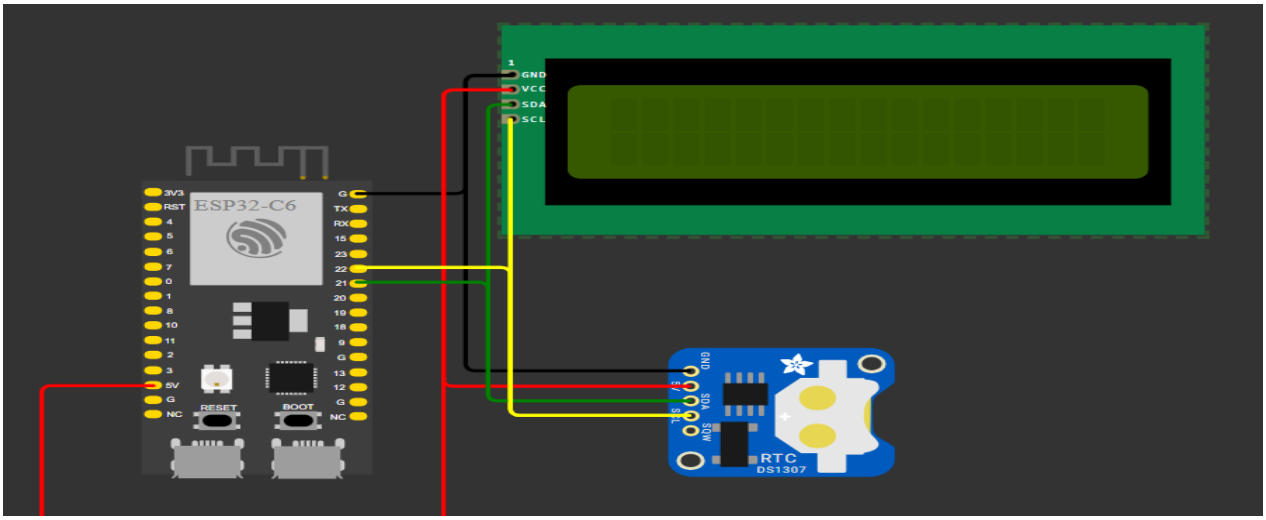


Εικόνα 60: Rain Sensor Pinout

MH-RD		
PIN	SYMBOL	CONN
1	A0	KENO
2	D0	GPIO23
3	GND	GND
4	VCC	3.3V
5	**Rain Board**	
6	**Rain Board**	

Πίνακας 4: Rain Sensor Connections

I2C MODULES (LCD-I2C INTERFACE & RTC DS1307)



Εικόνα 61: Σύνδεση I2C οθόνης και RTC στον ESP32

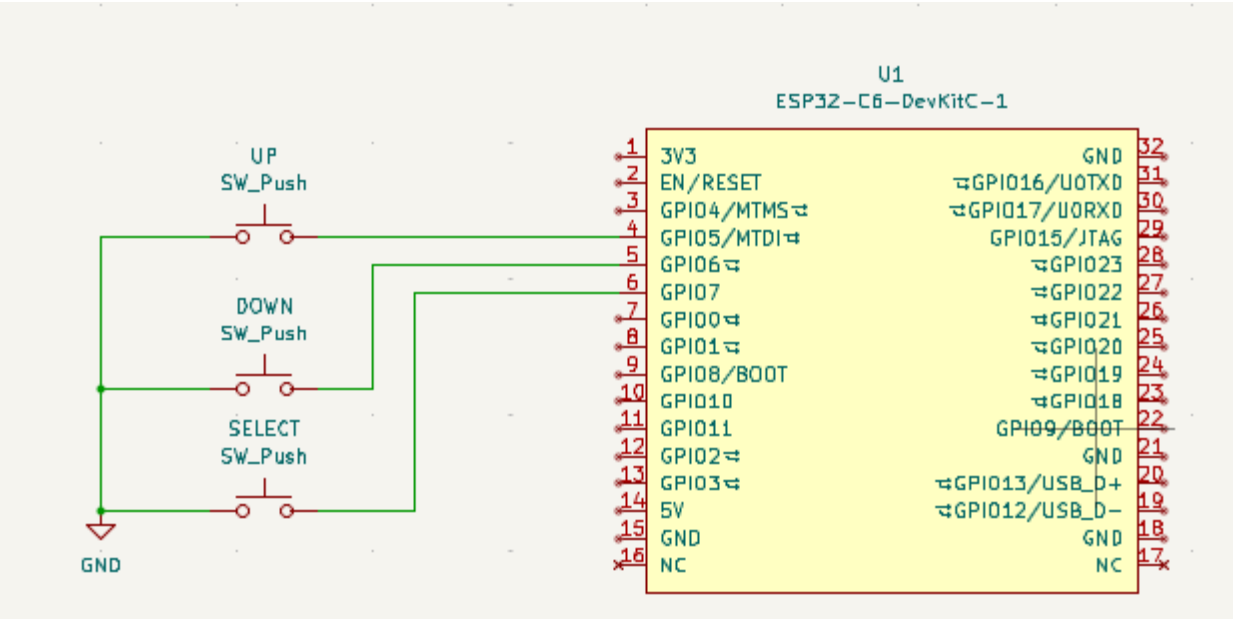
16X2 LCD & I2C INTERFACE MODULE		
PIN	SYMBOL	CONN
1	GND	GND
2	VCC	5V
3	SDA	GPIO21
4	SCL	GPIO22

DS1307		
PIN	SYMBOL	CONN
1	DS	KENO
2	SCL	GPIO22
3	SDA	GPIO21
4	VCC	5V
5	GND	GND

Πίνακας 5: RTC DS1307 Connections & LCD I2C

PUSH BUTTON CONNECTION

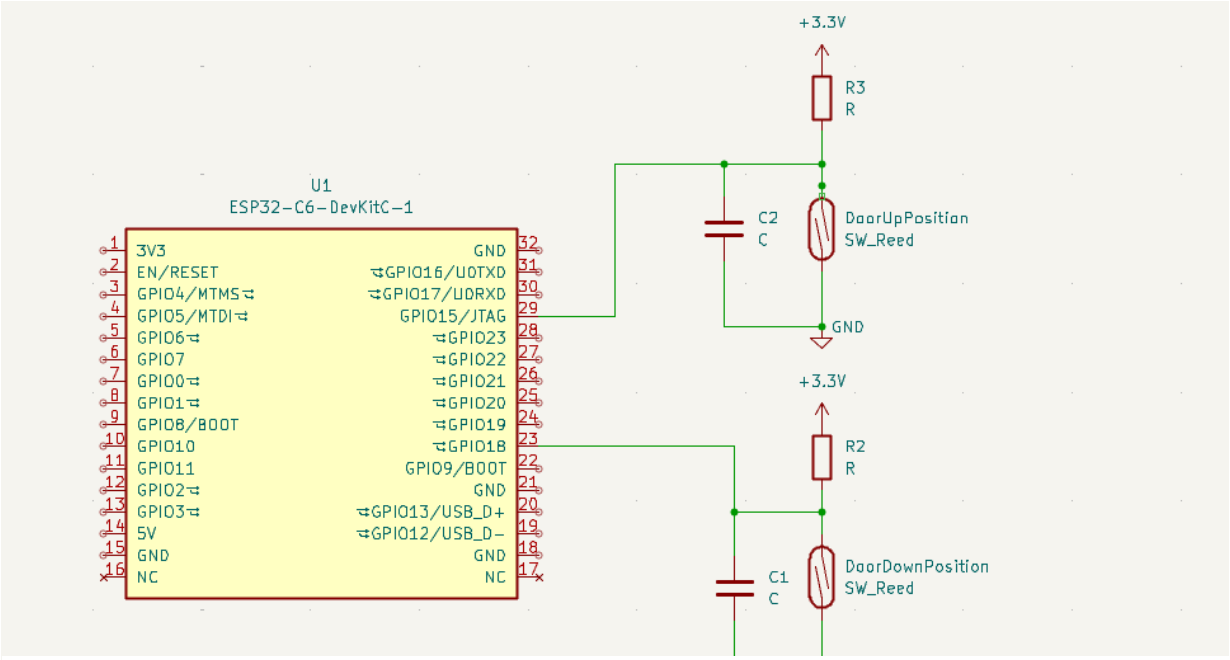
Σε αυτή την συνδεσμολογία λόγο του ότι χρησιμοποιούμε τις internal pull ups του Esp32 συνδέουμε άμεσα τα pin από τους διακόπτες στην γείωση και τον μικροελεκτη



Εικόνα 62: Συνδεσμολογία push-button στον ESP32

MC-38W ΜΑΓΝΗΤΙΚΟΣ ΔΙΑΚΟΠΤΗΣ

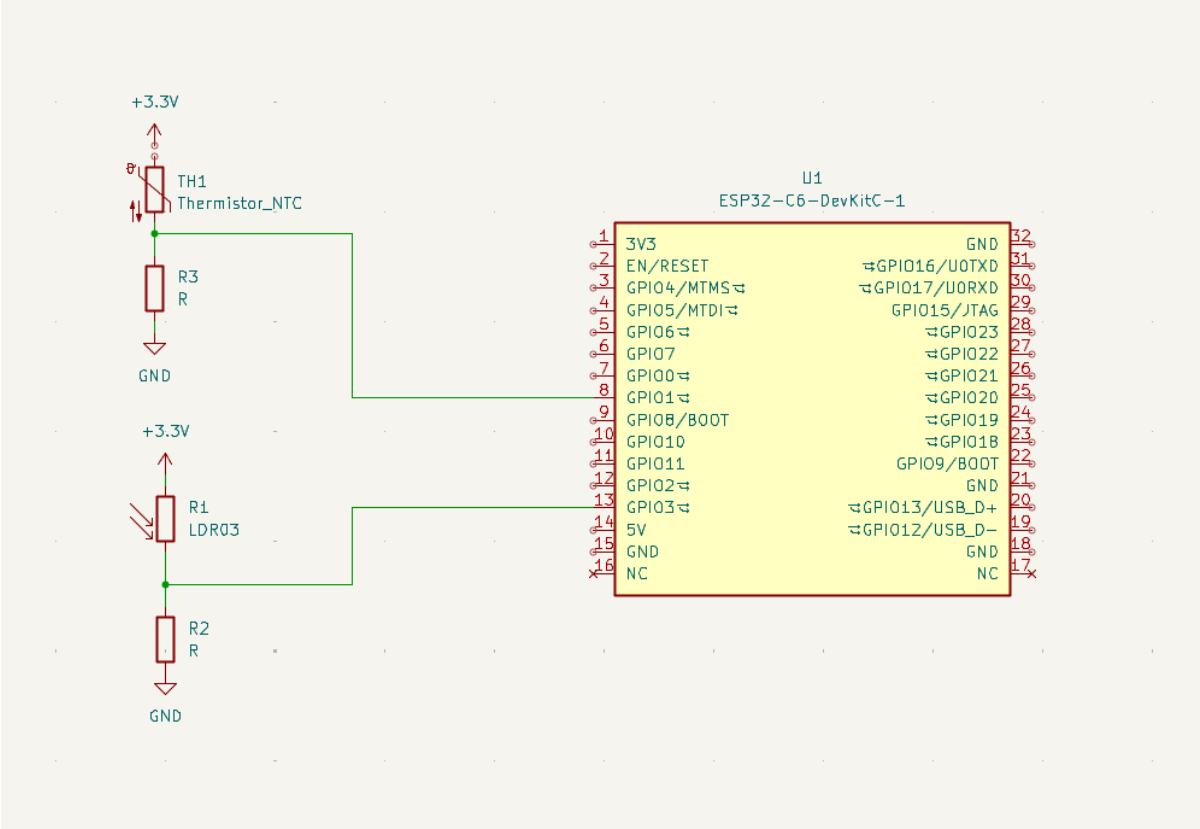
Σε αυτή την συνδεσμολογία χρησιμοποιούμε και εσωτερική και εξωτερική αντίσταση pull up καθώς και δυο πυκνωτές σε ρόλο πυκνωτών αποθολωσης επειδή υπάρχουν μακριά καλώδια ώστε να έχουμε όσο δυνατών πιο καθαρή είσοδο στον μικροελεκτη



Εικόνα 63: Συνδεσμολογια Μαγνητικών επαφών στον ESP32

THERMISTOR & PHOTORESISTOR

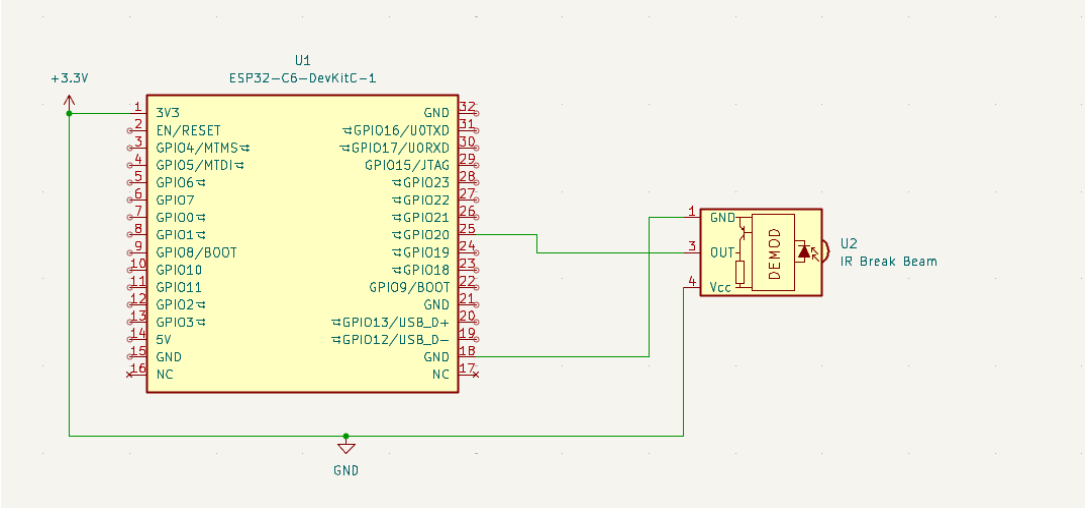
Χρησιμοποιούμε τα αναλογικά pin του esp32 0 και 3 για να συνδέσουμε την LDR και το θερμίστορ, με εξωτερικές αντιστάσεις pull-down έτσι διαβάζοντας την αναλογική είσοδο με τον κώδικα μπορούμε να καταλάβουμε την θερμοκρασία και την ποσότητα του φωτός που υπάρχει στο περιβάλλον.



Εικόνα 64: Συνδεσμολογία θερμιστορ & LDR στον ESP32

IR BREAK BEAM

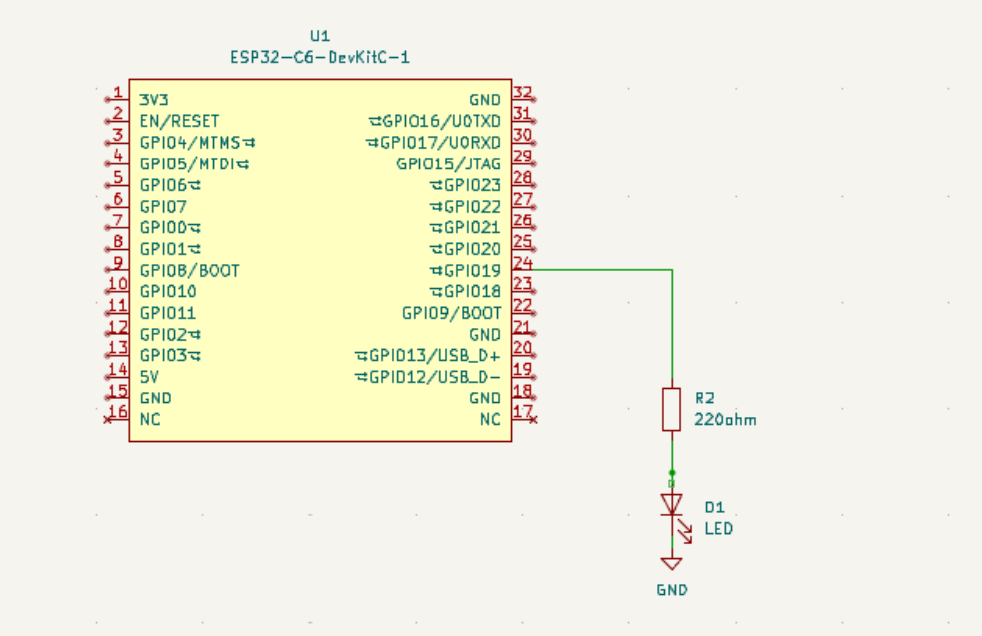
Σε αυτή την σύνδεση Χρησιμοποιείται ξανά η εσωτερική αντίσταση του μικροελεκτη



Εικόνα 65: Συνδεσμολογία IR Break Beam στον ESP32

LED ΠΡΟΕΙΔΟΠΟΙΗΣΗΣ

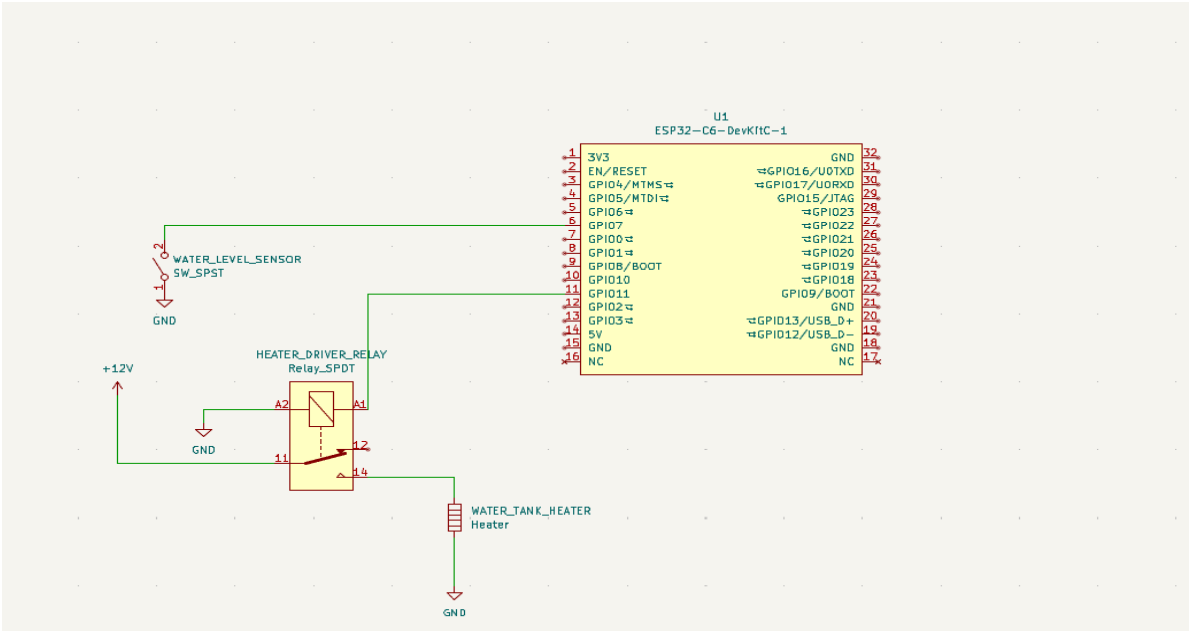
Το led αυτό συνδεδεμένο σε σειρά με μια αντίσταση 220Ω για να διασφαλίσουμε ότι δεν θα καεί από υπερβολικό ρεύμα είναι αυτό που μας ειδοποιεί οπτικά όταν υπάρχει λίγο φαγητό, λίγο νερό ή λείπει κάποια κότα την στιγμή που κλείσει η πόρτα



Εικόνα 66: Συνδεσμολογία Warning LED στον ESP32

WATER LEVEL SENSOR ΚΑΙ HEATER

Σε αυτή την συνδεσμολογία έχουμε τον αισθητήρα στάθμης νερού συνδεδεμένο στο GPIO7, χρησιμοποιούμε την εσωτερική pull-up του esp32 όπως επίσης και το θερμικό σώμα το οποίο δουλεύει σαν defrost στο νερό σε περίπτωση παγετού. Αυτό είναι συνδεδεμένο μέσω ενός Relay 3V οδηγούμενο από το GPIO11 δίνοντας έτσι στο θερμαντικό την απαραίτητη 12V τάση που χρειάζεται.



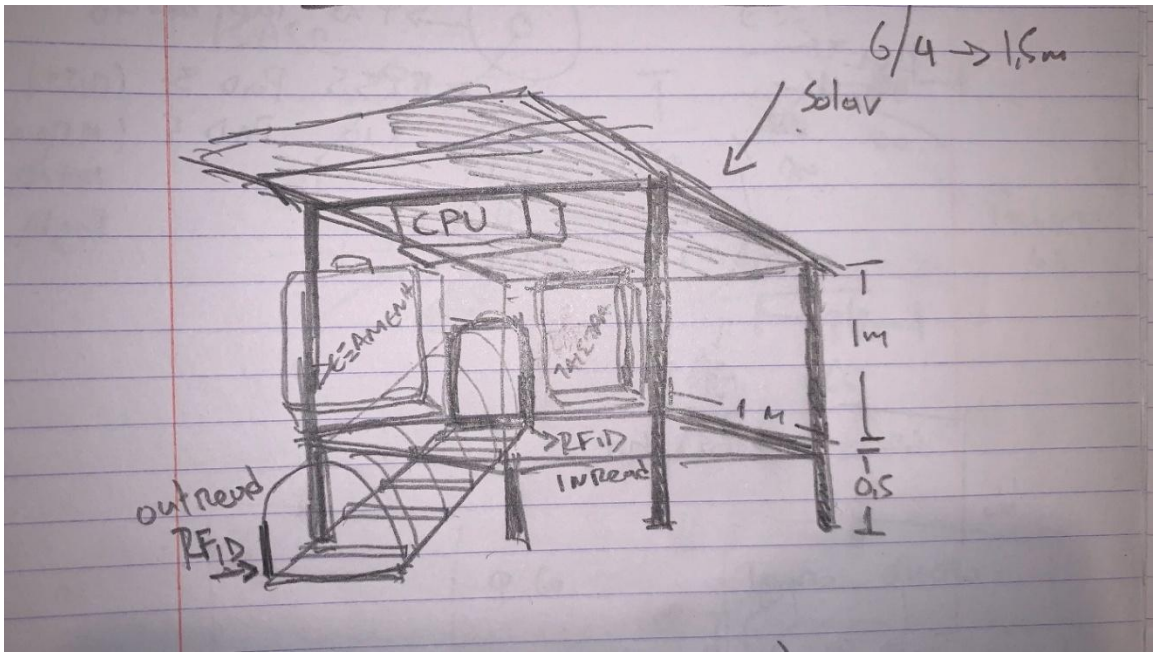
Εικόνα 67: Water Sensor & Water Heater

4.2 Κατασκευή Θαλάμου και Επιμέρους Τμημάτων

Η φυσική κατασκευή του θαλάμου αποτελεί ένα από τα πιο σημαντικά στάδια της υλοποίησης, καθώς εδώ συνδυάζονται όλα τα ηλεκτρονικά, μηχανικά και λειτουργικά στοιχεία σε ένα πραγματικό, λειτουργικό περιβάλλον. Ο σχεδιασμός του θαλάμου έγινε με γνώμονα τη **λειτουργικότητα, την εργονομία και την αντοχή στις συνθήκες περιβάλλοντος**, ενώ παράλληλα λήφθηκε υπόψη η ευκολία συντήρησης και επέκτασης του συστήματος. Ο θάλαμος κατασκευάστηκε με υλικά κατάλληλα για εξωτερική χρήση, ώστε να διασφαλίζεται η προστασία του εξοπλισμού και η ευημερία των ορνίθων. Η εσωτερική διαρρύθμιση σχεδιάστηκε με τρόπο ώστε να επιτρέπει την εύκολη τοποθέτηση και συντήρηση των αισθητήρων και περιφερειακών, καθώς και την απρόσκοπτη κίνηση των ζώων. Ιδιαίτερη έμφαση δόθηκε στην κατασκευή δύο βασικών κινητών μηχανισμών:

- **Η αυτόματη πόρτα**: Σχεδιάστηκε και εκτυπώθηκε σε 3D εκτυπωτή, με στόχο την αυτόνομη διαχείριση της πρόσβασης των ορνίθων εντός και εκτός του θαλάμου, με βάση τον φωτισμό ή την ώρα.
- **Η αυτόματη ταΐστρα**: Επίσης σχεδιασμένη και υλοποιημένη μέσω 3D εκτύπωσης, επιτρέπει την ακριβή και χρονοπρογραμματισμένη παροχή τροφής, προσφέροντας μεγαλύτερη αυτονομία στο σύστημα.

Κάθε ένα από αυτά τα τμήματα θα παρουσιαστεί αναλυτικά στις αντίστοιχες υποενότητες, με περιγραφή του σχεδιασμού, της εκτύπωσης, των υλικών που χρησιμοποιήθηκαν καθώς και του τρόπου ενσωμάτωσης των κινητήρων και αισθητήρων. Αυτή η προσέγγιση ανέδειξε τη δυνατότητα αξιοποίησης **τεχνολογιών κατασκευής όπως η 3D εκτύπωση**, προκειμένου να προσαρμοστεί το σύστημα στις συγκεκριμένες ανάγκες του έργου, μειώνοντας το κόστος και αυξάνοντας την ευελιξία.

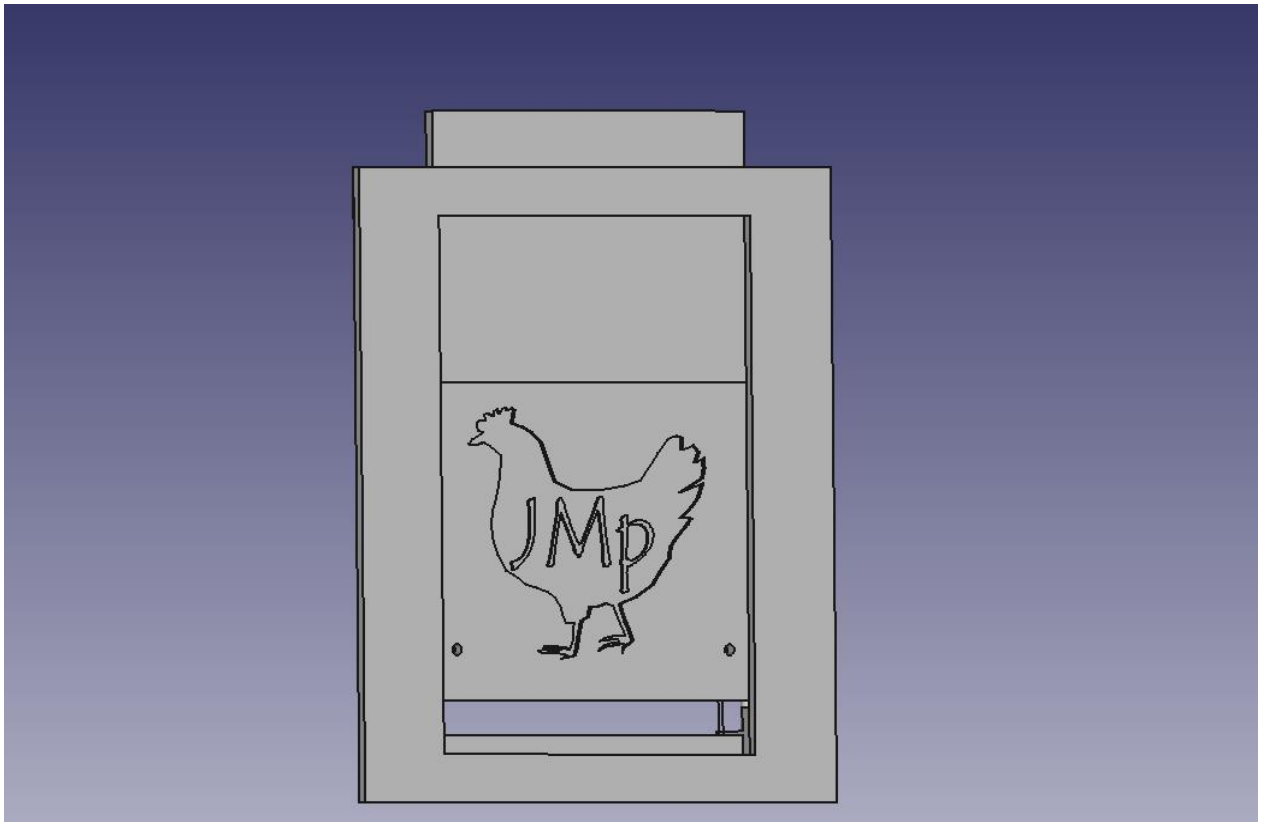


Εικόνα 68: Πρόχειρος σχεδιασμός Θαλάμου

4.2.1 Κατασκευή Πόρτας

Η αυτόματη πόρτα του θαλάμου αποτελεί ένα από τα πλέον ουσιαστικά υποσυστήματα, καθώς εξασφαλίζει τον αυτοματοποιημένο έλεγχο της πρόσβασης των ορνίθων, προσφέροντας ταυτόχρονα προστασία από εξωτερικές απειλές (όπως θηρευτές ή καιρικές συνθήκες) και έλεγχο του κύκλου εισόδου/εξόδου σύμφωνα με τις ρυθμίσεις του συστήματος. Η σχεδίαση της έγινε εξ ολοκλήρου με γνώμονα την πρακτικότητα, την αξιοπιστία και τη δυνατότητα ενσωμάτωσης κινητήρα, με όλα τα εξαρτήματα να έχουν παραχθεί με χρήση 3D εκτύπωσης.

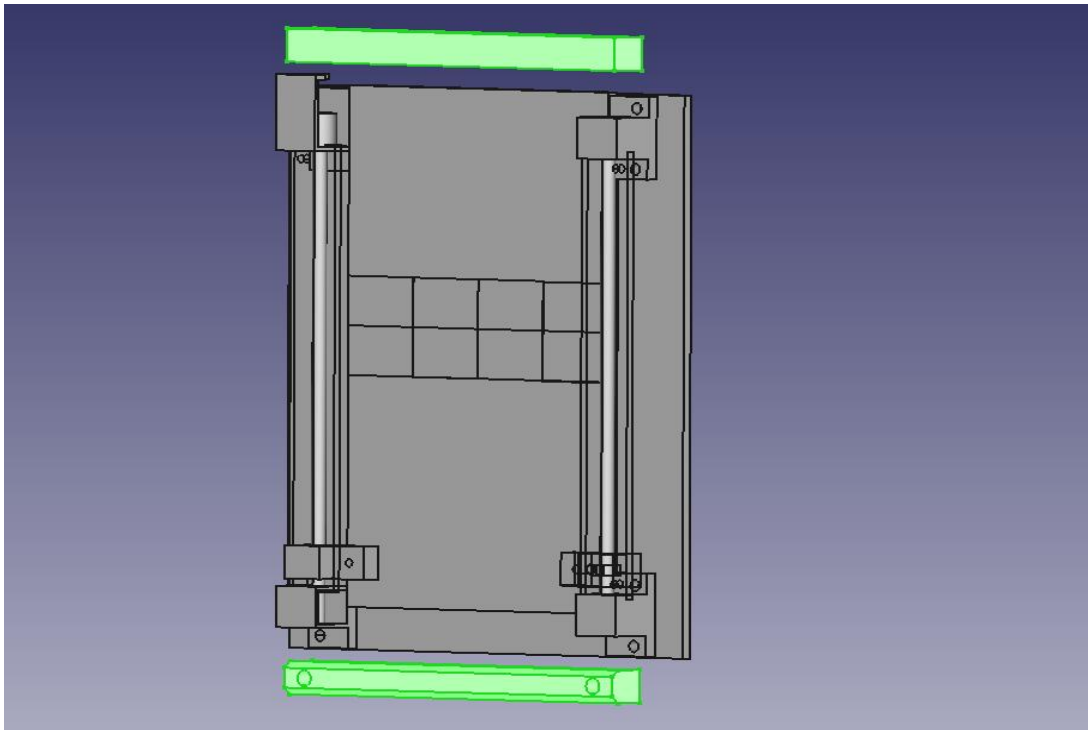
Βασικό σημείο της κατασκευής ήταν η λειτουργεία να πραγματοποιείτε με αξιόπιστο τρόπο να είναι φτιαγμένη από υλικό που να αντέχει σε δυσμενείς συνθήκες στην ηλιακή ακτινοβολία σε φθορά του χρόνου και να μην έχει μηχανικά μέρη που να μπορούν να πάθουν βλάβη εύκολα. Παρατηρώντας διάφορες πλατφόρμες όπως για παράδειγμα αυτή του bed από τον 3D εκτυπωτή Σκέφτηκα να σχεδιάσω μία πόρτα η οποία χρησιμοποιώντας ένα μοτέρ dc γυρνώντας έναν ατέρμονο άξονα θα σήκωνε μία βάση που θα είχε ενσωματωμένο ένα παξιμάδι μέσα και εκεί θα βίδωνε το τμήμα της πόρτας. από την άλλη πλευρά θα έχει έναν άξονα για οδηγό καθώς και ένα συμπαγές Πλαίσιο που θα χρησιμοποιηθεί για να στηρίξει την πόρτα και σαν έξτρα οδηγός ώστε να μην την αφήνει να κάνει κινήσεις που θα την βγάλουν από την θέση της.



Εικόνα 69: Ολοκληρωμένη Πόρτα Θαλάμου

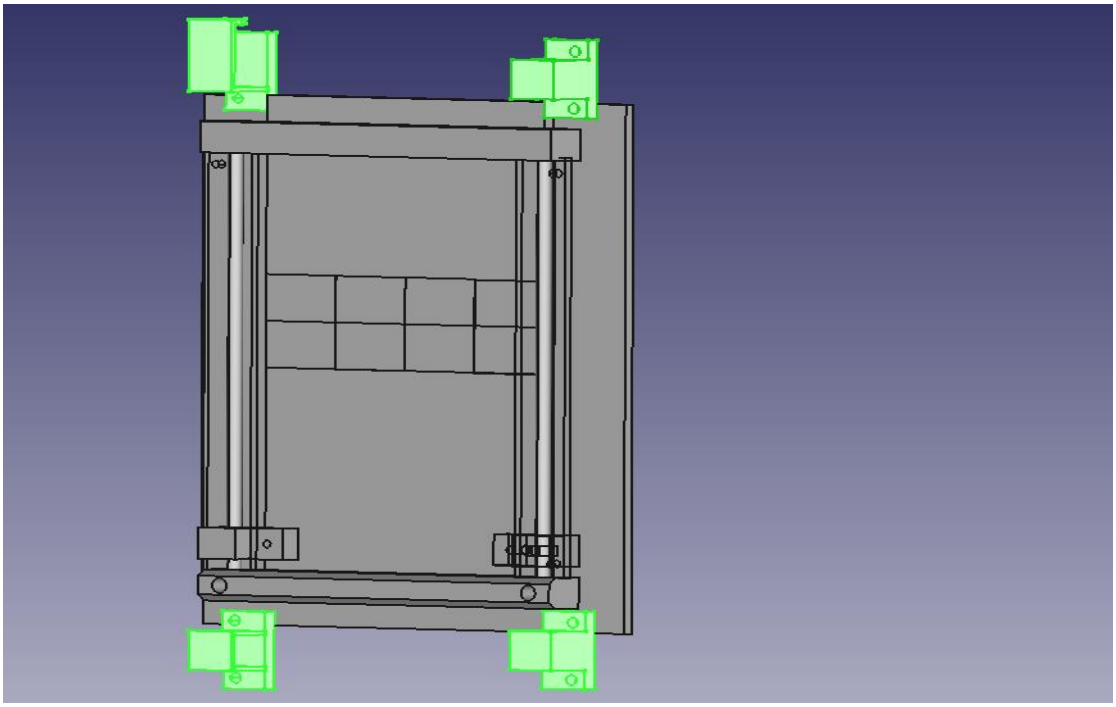
Παρακάτω λοιπόν θα δούμε τμηματικά τα στάδια σχεδιασμού και εκτύπωσης της πόρτας

4.2.1.1 Βραχίονες Στήριξης και Βάσεις Στερέωσης



Εικόνα 70: Βραχίονες στήριξης

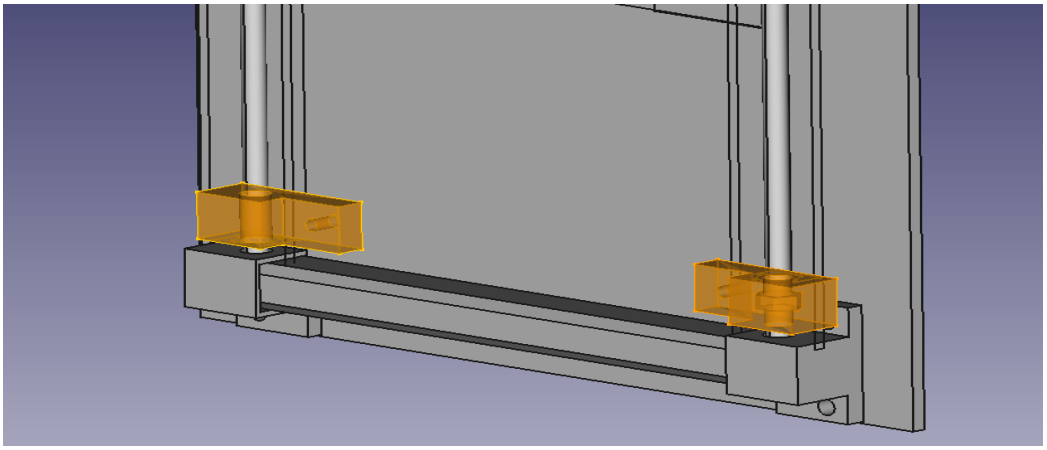
Οι βραχίονες στήριξης αποτελούν τους βασικούς φορείς μηχανικής ενίσχυσης της κατασκευής, καθώς κρατούν και ευθυγραμμίζουν τον κάθετο άξονα και τα κινούμενα τμήματα της πόρτας. Μαζί με τις βάσεις στερέωσης, που προσδένουν τη συνολική δομή στο πλαίσιο του θαλάμου, διασφαλίζουν τη σταθερότητα και την απορρόφηση των κραδασμών κατά την κίνηση.



Εικόνα 71: Βάσεις στήριξης

4.2.1.2 Άξονες και Σύστημα Ανύψωσης

Η πόρτα κινείται κατακόρυφα μέσω οδηγών-αξόνων, πάνω στους οποίους γλιστράει η ίδια η κατασκευή. Οι άξονες αποτελούνται από έναν άξονα ατέρμονα Φ8 και έναν λείο Φ8 μήκους 30εκατοστων. Οδηγούν την κίνηση με ακρίβεια, χωρίς να απαιτούν ιδιαίτερη συντήρηση. Το σύστημα περιλαμβάνει επίσης τις **βάσεις ανύψωσης**, οι οποίες μεταφέρουν την κίνηση από τον κινητήρα προς την πόρτα, επιτρέποντας το άνοιγμα και κλείσιμο με σταθερή ταχύτητα και χωρίς απότομες κινήσεις.



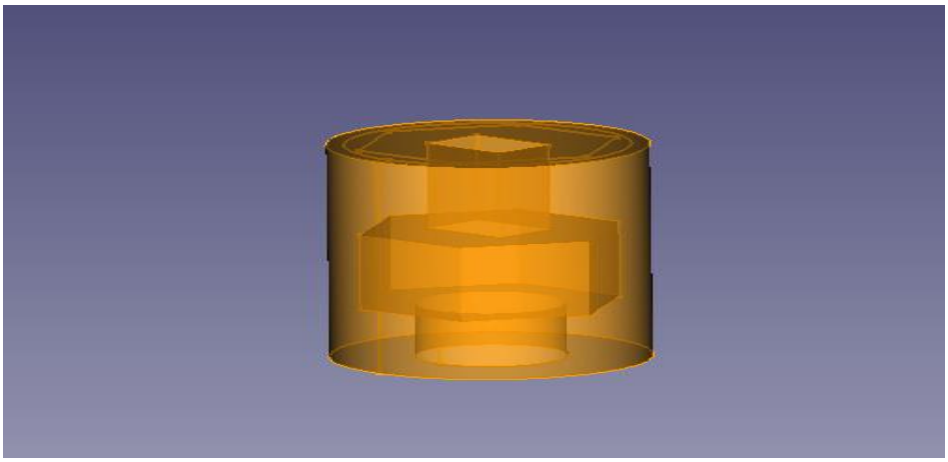
Εικόνα 72: Σύστημα Ανύψωσης

στην μια πλευρά σε αυτή του ατέρμονα άξονα μέσα στην βάση ανύψωσης ενσωματώθηκε παξιμάδι ώστε γυρνώντας ο άξονας σε σταθερό σημείο αναγκάζει την βάση να κινείτε προς τα πάνω η προς τα κάτω

.

4.2.1.3 Αντάπτορας Μοτέρ και Μετάδοση Κίνησης

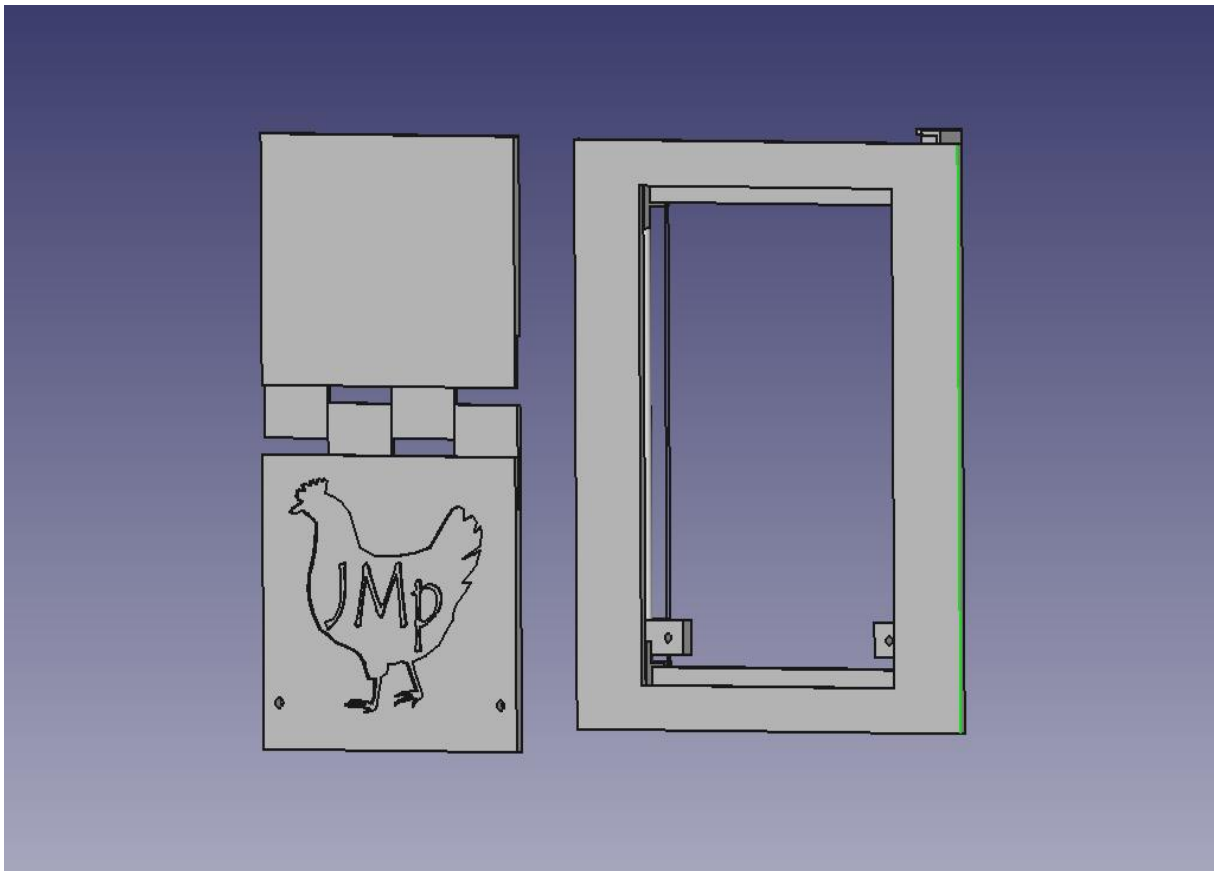
Για τη σύνδεση του μοτέρ με το μηχανισμό κίνησης της πόρτας, χρησιμοποιήθηκε ένας **αντάπτορας**, ο οποίος προσαρμόστηκε επακριβώς στον άξονα του μοτέρ και κουμπώνει με τις βάσεις της πόρτας. Αυτό το εξάρτημα λειτουργεί ως μεταβατικός κρίκος που μεταφέρει ροπή με ακρίβεια, αποτρέποντας ολισθήσεις ή μηχανικές απώλειες. Ο αντάπτορας αυτός Είναι ένα από τα σημαντικότερα τμήματα του σχεδιασμού της πόρτας και για λόγους ενισχύσεις και αντοχής ενσωματώθηκε ένα παξιμάδι οπού εκεί βιδώνει ο ατέρμον άξονας και στην συνέχεια κολλήθηκε με κόλλα σπειρωμάτων.



Εικόνα 73: Αντάπτορας Ένωσης Μοτέρ Με Άξονα

4.2.1.4 Πλαίσιο Πόρτας και Οδηγοί Κίνησης

Το πλαίσιο διαμορφώνει τα όρια κίνησης της πόρτας και εξασφαλίζει την ομαλή κύλιση της, λειτουργώντας παράλληλα και ως μηχανισμός ασφάλειας. Σχεδιάστηκε με ανοχές που επιτρέπουν την κίνηση χωρίς μπλοκαρίσματα αλλά και περιορίζουν τη σκόνη και τα σωματίδια να εισχωρούν στα εσωτερικά κινούμενα μέρη.



Εικόνα 74: Πλαίσιο και Κεντρική Επιφάνεια Πόρτας

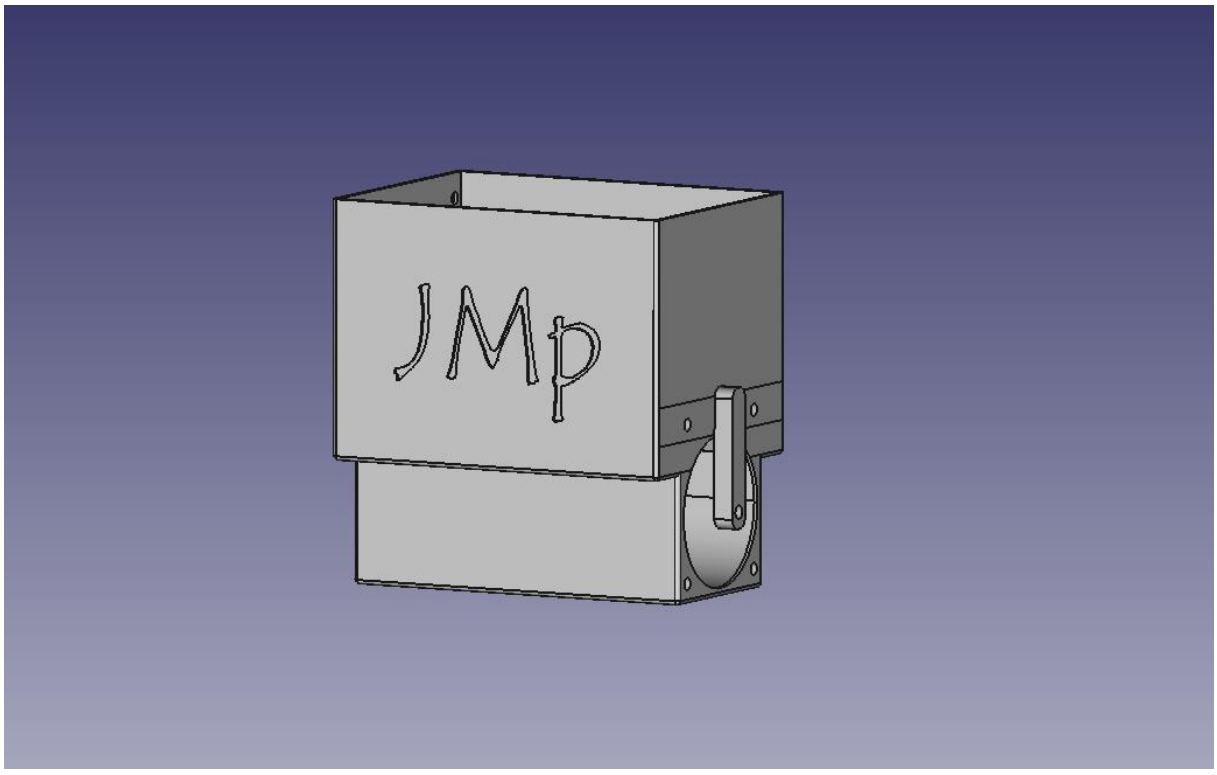
4.2.1.5 Η Κεντρική Επιφάνεια της Πόρτας

Η ίδια η πόρτα κατασκευάστηκε από συμπαγές, ελαφρύ πλαστικό και προσαρμόστηκε έτσι ώστε να είναι αρκετά ανθεκτική αλλά ταυτόχρονα εύκολη στην ανύψωση από το μοτέρ. Το σχήμα της επιτρέπει πλήρες κλείσιμο του ανοίγματος, χωρίς κενά που μπορεί να επιτρέψουν τη διαφυγή ορνίθων ή την είσοδο ανεπιθύμητων παραγόντων.

Η πόρτα σχεδιαστικέ και εκτυπώθηκε σε δυο τμήματα διότι η επιφάνεια εκτύπωσης του συγκεκριμένου εκτυπωτή που χρησιμοποιήθηκε δεν επέτρεπε ένα ενιαίο κομμάτι. Παρόλα αυτά με τις κατάλληλες εγκοπές η πόρτα ενώνεται και κολλητέ και προσφέρει την κατάλληλη ασφάλεια και αντοχή.

4.2.2 Κατασκευή Ταΐστρας

Η αυτόματη ταΐστρα αποτελεί ένα από τα πιο κρίσιμα υποσυστήματα του αυτόματου θαλάμου, καθώς είναι υπεύθυνη για τη διανομή τροφής με ελεγχόμενο και προγραμματισμένο τρόπο. Ο σχεδιασμός της βασίστηκε στην ανάγκη για **ακρίβεια, αξιοπιστία και δυνατότητα ενσωμάτωσης με τον μικροελεγκτή ESP32**, ενώ η κατασκευή της πραγματοποιήθηκε εξ ολοκλήρου με 3D εκτυπωτή, ώστε να προσαρμοστεί πλήρως στις απαιτήσεις του έργου. Η κατασκευή της ταΐστρας παρόλο που αποτελείται από λιγότερα μέρη από ότι η πόρτα ήτανε αρκετά πιο δύσκολη σαν κατασκευή. αυτό γιατί ο τρόπος που έπρεπε να σχεδιαστεί ήταν τέτοιος ώστε το μοτέρ να μη ζορίζεται κατά τη διάρκεια του ταΐσματος , η χωρητικότητα της να είναι αρκετή ώστε να έχει αρκετή αυτονομία και το φαγητό να μην φρακάρει είτε στα τοιχώματα είτε στην έξοδο . Αυτός ο συνδυασμός για να επιτευχθεί θα έπρεπε να είναι έτσι ώστε να μην χρειάζεται μεγάλο μοτέρ για να έχει δύναμη να γυρνάει και να ρίχνει την τροφή και η ταΐστρα να έχει τέτοιο σχήμα και κλήσεις που θα επέτρεπαν την τροφή να πέφτει χωρίς πρόβλημα. έτσι κατέληξα στην χρήση κοχλία όπου γυρνάει από το μοτέρ DC,. Για να αποφευχθούν κολλήματα του κοχλία μέσο κώδικα και του driver πραγματοποιεί κάποιες ανάποδες στροφές και έτσι διασφαλίζεται η ομαλή τροφοδοσία της τροφής. Στο εσωτερικό της ταΐστρας έχουμε ενσωματωμένο τον αισθητήρα υπέρυθρης ακτίνας ο οποίος τελειώνοντας η τροφή αποκτά επικοινωνία με το transmitter του και έτσι μας στέλνει την κατάλληλη ειδοποίηση.



Εικόνα 75: ολοκληρωμένη Μορφή Ταΐστρας

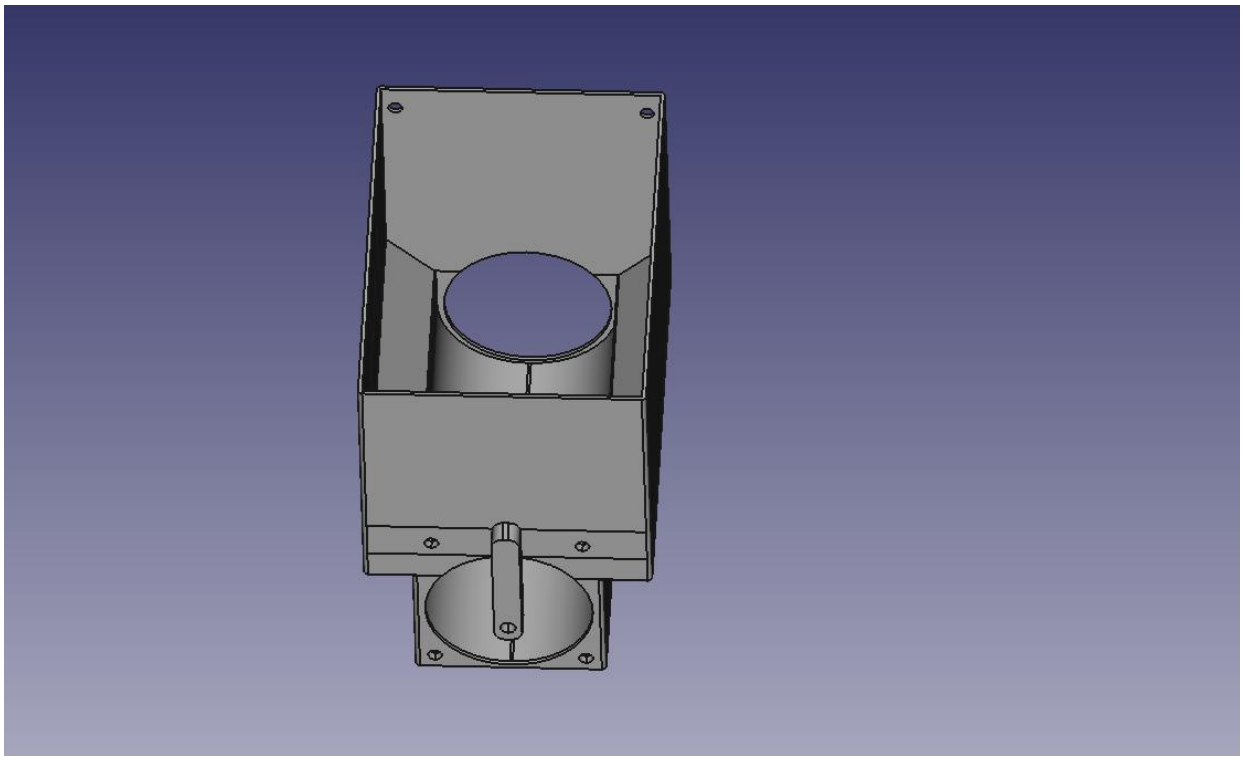
Παρακάτω θα δούμε τμηματικά την κατασκευή της στο σχεδιαστικό πρόγραμμα.

Η ταΐστρα αποτελείται από 3 βασικά μέρη. Το κύριο σώμα, τον κοχλία και την βάση του μοτέρ που ενώνει όλα τα υπόλοιπα κομμάτια. Υπάρχουν και κάποια περαιτέρω κομμάτια όπως το καπάκι για να κλείνει από πάνω καθώς και επέκταση του ύψους της ταΐστρας σε περίπτωση που θέλουμε εξτρά ποσότητα φαγητού.

4.2.2.1 Κύριο Σώμα Ταΐστρας

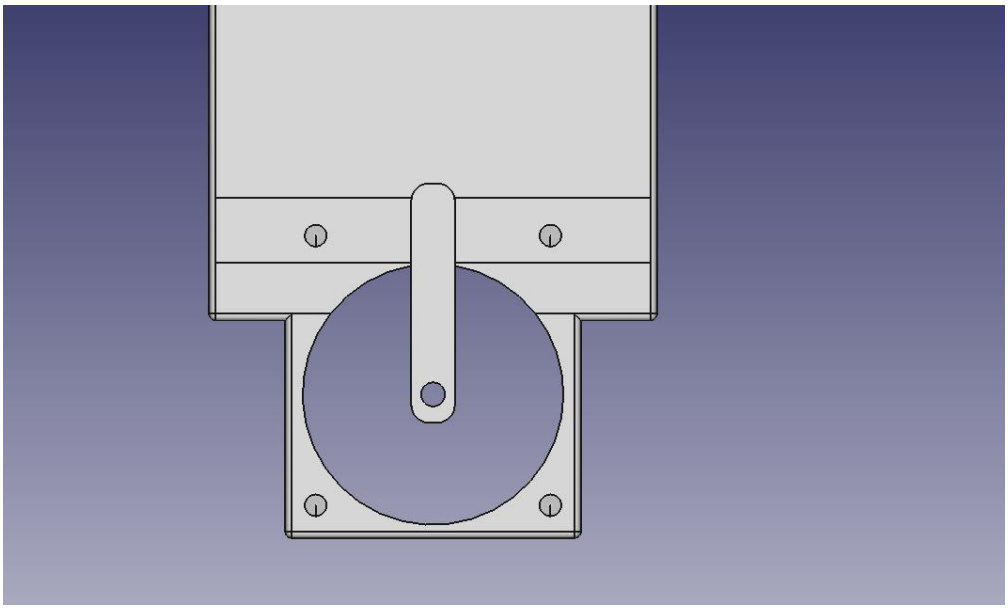
Το κύριο σώμα της ταΐστρας αποτελεί τον βασικό αποθηκευτικό χώρο για την τροφή. Σχεδιάστηκε έτσι ώστε να έχει επαρκή χωρητικότητα για την κάλυψη της διατροφής των ορνίθων για παρατεταμένο χρονικό διάστημα, μειώνοντας την ανάγκη για συνεχή ανθρώπινη επίβλεψη.

- **Υλικό Κατασκευής**:ASA πλαστικό, μέσω 3D εκτύπωσης, επιλεγμένο για την ανθεκτικότητά του σε συνήθεις εξωτερικές συνθήκες (θερμοκρασία/υγρασία).
- **Σχήμα**: Αποτελείται από ένα δοχείο το οποίο έχει τις κατάλληλες κλήσεις προκειμένου να φτάνει η τροφή ευκολά από το βάρος της στο σημείο οπού βρίσκεται ο κοχλίας. Έχει δυο υποδοχές όπου κουμπώνουν τα Ir Break Beam ώστε να έχουμε εικόνα της ποσότητας του φαγητού που βρίσκεται μέσα. Έχει χωρητικότητα περίπου 2κιλων φαγητού χωρίς την προέκταση.
- **Άνοιγμα πλήρωσης**: Σχεδιασμένο για εύκολο ανεφοδιασμό χωρίς ανάγκη αποσυναρμολόγησης.



Εικόνα 76 : Κύριο Σώμα Ταΐστρας

Στο σημείο που εφαρμόζει ο κοχλίας υπάρχει ιδική υποδοχή που τον κρατάει σταθερό από την μια πλευρά επιτρέποντας του όμως να περιστρέφεται με ευκολία. Στην πίσω πλευρά είναι ανοιχτό και με βίδες προσαρμόζεται η βάση του μοτέρ η οποία και κρατάει όλο το σύστημα στην θέση του. Έχοντας αυτό τον σχεδιασμό έχουμε την δυνατότητα να προσαρμόσουμε κατάλληλο κοχλία ανάλογα το είδος και την υφή της τροφής καθώς επίσης και να αντικαταστήσουμε τον κοχλία σε περίπτωση φθοράς.

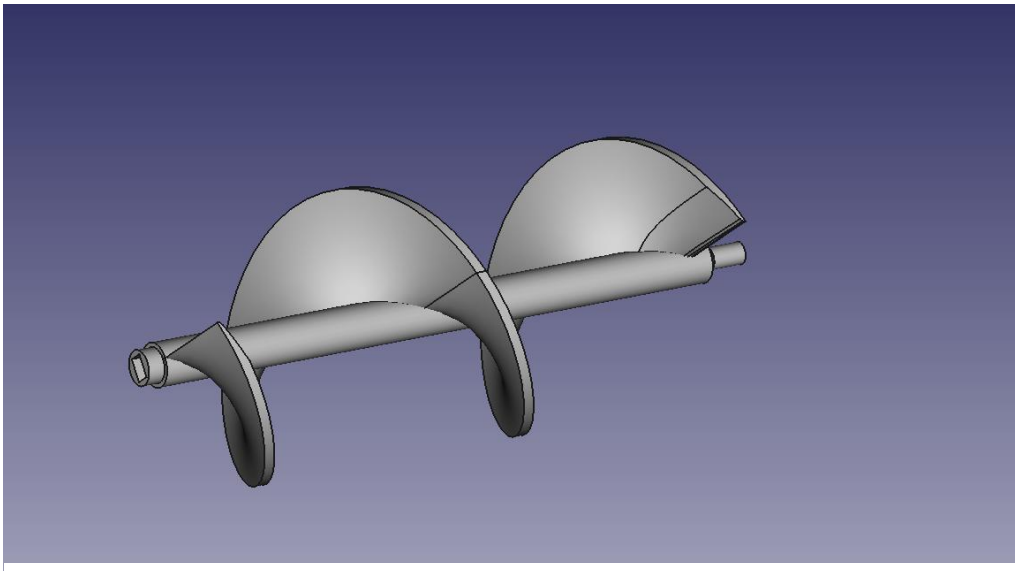


Εικόνα 77: Υποδοχή Κοχλία

4.2.2.2 Κοχλίας Διανομής Τροφής

Στην καρδιά της λειτουργίας της ταΐστρας βρίσκεται ο **κοχλίας (auger screw)**, ένας περιστρεφόμενος μηχανισμός που μεταφέρει την τροφή από τον αποθηκευτικό χώρο προς το σημείο εξόδου.

- **Αρχή Λειτουργίας**: Κατά την περιστροφή του, ο κοχλίας σπρώχνει την τροφή μπροστά μέσα από το σπειροειδές του σώμα, με σταθερή και ελεγχόμενη ροή.
- **Σχεδίαση**: Εκτυπωμένος σε ASA, με ειδική μέριμνα για την ακρίβεια του βήματος του σπειρώματος και τη σταθερή εφαρμογή στον άξονα του μοτέρ.
- **Συνεργασία με τον μικροελεγκτή**: Η λειτουργία του κοχλία ενεργοποιείται μέσω εντολής από τον ESP32, βάσει ώρας ή μέσω trigger.κατά την λειτουργία του πραγματοποιεί και ανάποδες στροφές ώστε ξεκολλήσει τυχών τροφή αλλά και μέσω τον δονήσεων που προκαλεί αυτή η διαδικασία να την βοηθήσει να πέσει από το σημείο τροφοδοσίας,



Εικόνα 78: Κοχλίας

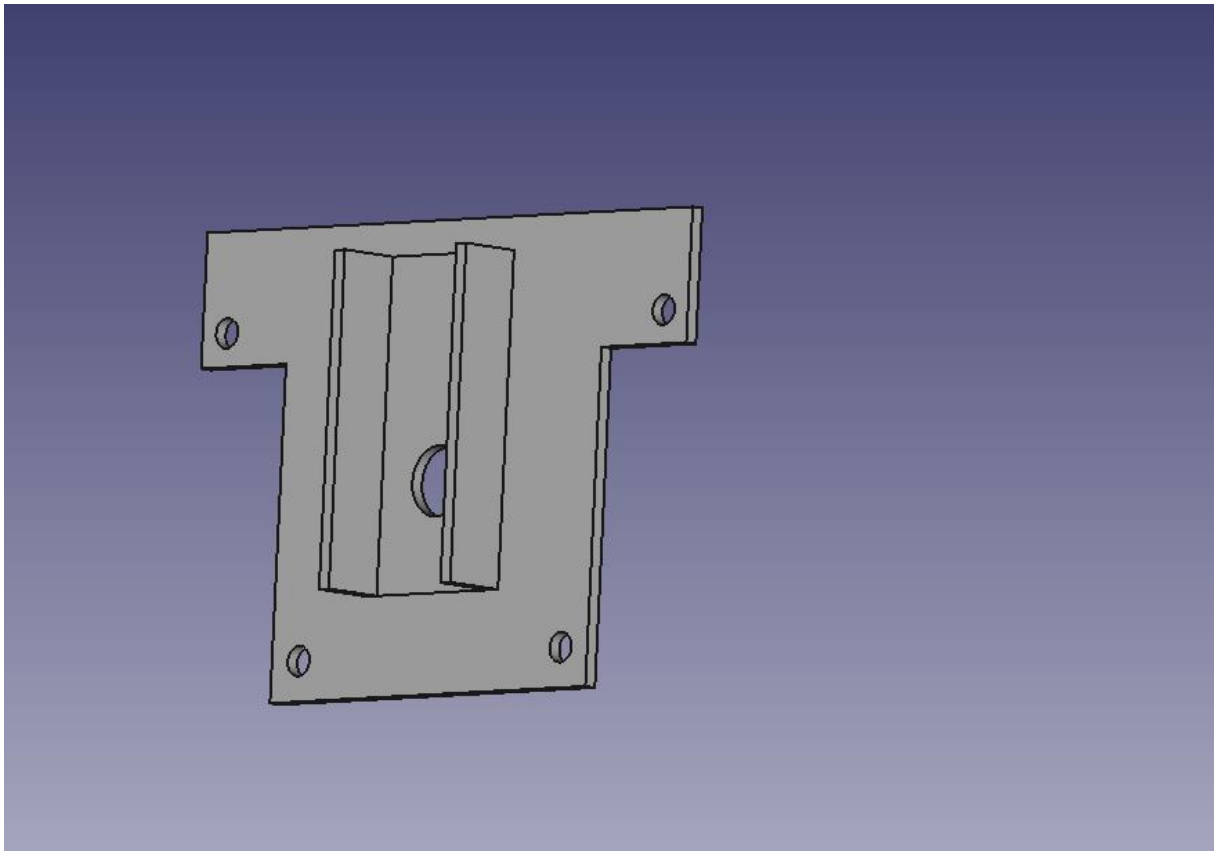
4.2.2.3 Βάση Μοτέρ και Στήριξη Μηχανισμού

Η σωστή στήριξη του μοτέρ αποτελεί κρίσιμο στοιχείο για τη λειτουργία της αυτόματης ταΐστρας, καθώς εξασφαλίζει τη μετάδοση της κίνησης προς τον κοχλία με ακρίβεια και σταθερότητα. Η βάση του μοτέρ σχεδιάστηκε ειδικά για να προσαρμόζεται τόσο στο σχήμα του κινητήρα όσο και στο εσωτερικό της κατασκευής, με τρόπο που να επιτρέπει ασφαλή στήριξη αλλά και εύκολη πρόσβαση για συντήρηση ή αντικατάσταση.

Η εκτύπωση της βάσης πραγματοποιήθηκε με 3D εκτυπωτή, χρησιμοποιώντας **υλικό ASA**, το οποίο παρέχει επαρκή μηχανική αντοχή και ακαμψία για τη συγκεκριμένη εφαρμογή. Η γεωμετρία της βάσης περιλαμβάνει εγκοπές και οδηγούς, ώστε να **ευθυγραμμίζεται απόλυτα με τον κοχλία** και να αποτρέπει την αστοχία στη μετάδοση ροπής, που θα μπορούσε να οδηγήσει σε ανώμαλη ή αναποτελεσματική λειτουργία του μηχανισμού διανομής τροφής.

Επιπλέον, έχουν προβλεφθεί **οπές στερέωσης** ώστε η βάση να μπορεί να βιδωθεί ή να κουμπώσει με ασφάλεια στο κύριο σώμα της ταΐστρας, περιορίζοντας τους κραδασμούς που ενδέχεται να μεταφερθούν στο πλαστικό σώμα κατά τη διάρκεια της λειτουργίας. Σε επίπεδο λειτουργικότητας, ο σχεδιασμός της βάσης επιτρέπει επίσης την **εύκολη αποσυναρμολόγηση του μοτέρ**, χωρίς να χρειάζεται η αποσύνδεση ολόκληρης της ταΐστρας – κάτι που είναι πολύτιμο για εργασίες συντήρησης ή αλλαγής εξαρτημάτων.

Τέλος, η βάση συμβάλλει και στην **καλύτερη θερμική διαχείριση** του μοτέρ, καθώς φέρει ανοίγματα που επιτρέπουν τη ροή αέρα, αποτρέποντας υπερθέρμανση σε περιβάλλοντα με παρατεταμένη λειτουργία.



Εικόνα 79: Βάση Στήριξης Μοτέρ

4.2.3 Κατασκευή Πλαισίων

Η κατασκευή των πλαισίων αποτελεί το δομικό υπόβαθρο του αυτόματου θαλάμου, εξασφαλίζοντας τη σταθερότητα, την ασφάλεια και τη μακροχρόνια αντοχή του συστήματος. Για την επίτευξη αυτών των στόχων, επιλέχθηκαν υλικά κατάλληλα για χρήση σε εξωτερικό περιβάλλον και σε επαφή με ζωντανούς οργανισμούς, όπως οι όρνιθες.

Εικόνα 80: Τα Βασικά υλικά Κατασκευής

Ο βασικός σκελετός κατασκευάστηκε με **γωνιακό σίδερο 20x20x3 mm**, το οποίο προσφέρει υψηλή μηχανική αντοχή ενώ παραμένει αρκετά ελαφρύ για ευκολότερη συναρμολόγηση και μεταφορά. Οι συνδέσεις των στοιχείων έγιναν με βίδωμα, διασφαλίζοντας τη στιβαρότητα , τη μακροχρόνια σταθερότητα του πλαισίου και την ευκολία στην συναρμολογηση.

Εικόνα 81: Οι Γωνιες Κατα τη Προετιμασία Κοπής

Εσωτερικά, για την πλήρωση των πλευρικών επιφανειών, χρησιμοποιήθηκαν **κοιλοδοκοί 20x20x2 mm** ώστε να σχηματιστούν υποστηρικτικές δομές που να διευκολύνουν την τοποθέτηση προστατευτικού πλέγματος και εξαρτημάτων. Το σύστημα καλύφθηκε με **πλέγμα τύπου "κουνελιού" διαστάσεων 1.25 x 1.25 cm**, το οποίο εξασφαλίζει επαρκή αερισμό και φωτισμό στο εσωτερικό του θαλάμου, ενώ παράλληλα προστατεύει τα πτηνά από εξωτερικές απειλές (π.χ. αρπακτικά ή τρωκτικά).

Εικόνα 82: Τα Πλαίσια Ολοκληρωμένα

Η επιλογή των υλικών έγινε με γνώμονα τόσο την αντοχή στις καιρικές συνθήκες όσο και την ασφάλεια των ζώων, αποφεύγοντας αιχμηρές ακμές και επιβλαβή υλικά. Το τελικό αποτέλεσμα είναι ένα πλαίσιο λειτουργικό, στιβαρό και εύκολα προσαρμόσιμο σε μελλοντικές επεκτάσεις ή βελτιώσεις

.

Εικόνα 83: Ο Θάλαμος Πριν Την Τοποθέτηση Των Ηλεκτρονικών

4.3 Συναρμολόγηση

Η φάση της συναρμολόγησης αποτέλεσε το τελικό και καθοριστικό στάδιο για τη μετάβαση από τα επιμέρους κατασκευαστικά στοιχεία σε ένα ολοκληρωμένο και λειτουργικό σύστημα. Σε αυτό το στάδιο πραγματοποιήθηκε η ένωση όλων των τμημάτων –μηχανικών και ηλεκτρονικών– με στόχο τη δημιουργία ενός πλήρως επιχειρησιακού αυτόματου θαλάμου.

Αρχικά τοποθετήθηκε ο βασικός σκελετός του θαλάμου, ο οποίος είχε κατασκευαστεί από μεταλλικά πλαίσια, και ενισχύθηκε με τις κατάλληλες βάσεις για τη στήριξη των επιμέρους μονάδων.

Εικόνα 84: Σύνδεση Των Επιμέρους Πλαίσιων

Στη συνέχεια, ενσωματώθηκαν τα μηχανικά συστήματα, όπως η αυτόματη πόρτα και η ταΐστρα, με προσοχή στην ευθυγράμμιση και την σταθερότητα των κινούμενων μερών.Παράλληλα, πραγματοποιήθηκε η εγκατάσταση της κεντρικής ηλεκτρονικής μονάδας, της καλωδίωσης και των αισθητήρων, σύμφωνα με το σχεδιάγραμμα της συνδεσμολογίας. Εγινε έλεγχος όλων των λειτουργιών και πραγματοποιήθηκαν οι πρώτες δοκιμές για τη βεβαίωση της ορθής συνεργασίας των επιμέρους μονάδων (κινητήρες, αισθητήρες, οθόνη, τροφοδοσία, συνδεσιμότητα).

4.4 Επίλογος Κεφαλαίου

Το τέταρτο κεφάλαιο ανέδειξε τη μετάβαση από τον θεωρητικό σχεδιασμό στην πρακτική υλοποίηση ενός πλήρους αυτόματου θαλάμου. Καταγράφηκαν αναλυτικά τα στάδια κατασκευής τόσο της κεντρικής μονάδας και των ηλεκτρονικών εξαρτημάτων, όσο και των επιμέρους μηχανικών τμημάτων του θαλάμου, όπως η πόρτα, η ταΐστρα και τα μεταλλικά πλαίσια στήριξης.

Η χρήση τεχνολογιών όπως η τρισδιάστατη εκτύπωση, η προσεκτική επιλογή υλικών και η ενσωμάτωση αυτοματισμών κατέστησαν εφικτή την κατασκευή ενός λειτουργικού και προσαρμόσιμου περιβάλλοντος για την εκτροφή ορνίθων, με δυνατότητα επεκτάσεων και βελτιώσεων στο μέλλον.

Κεφάλαιο 5ο: Λειτουργία Συστήματος και Απομακρυσμένος Έλεγχος

5.1 Εισαγωγή

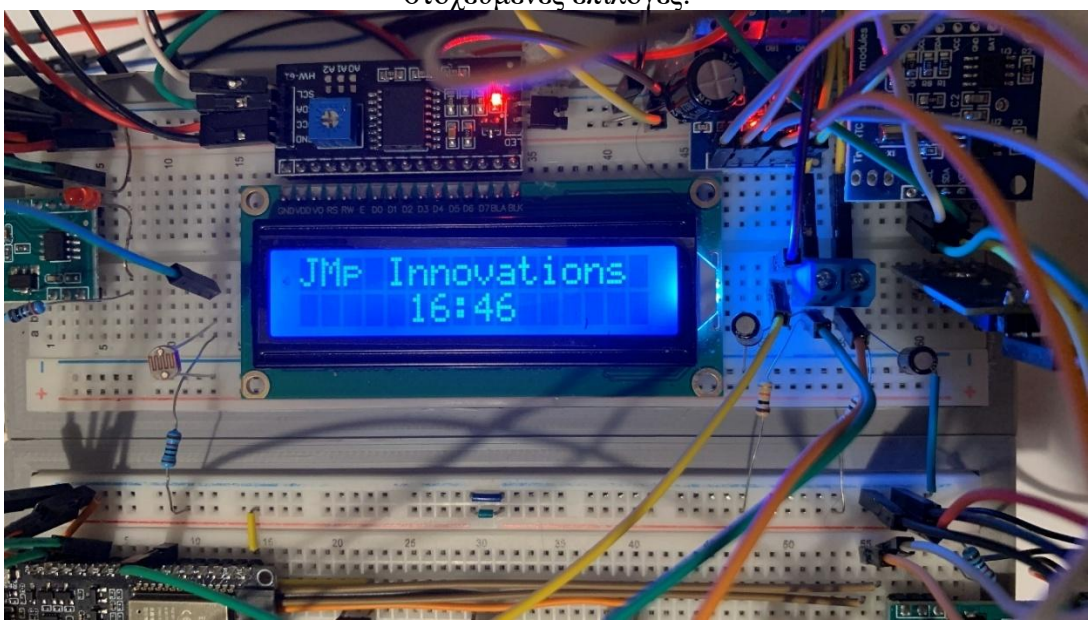
Το παρόν κεφάλαιο εστιάζει στη λειτουργική συμπεριφορά του αυτόματου θαλάμου στο πεδίο, αναδεικνύοντας τόσο τον τρόπο αλληλεπίδρασης του χρήστη με το σύστημα σε τοπικό επίπεδο μέσω της κεντρικής μονάδας, όσο και τις δυνατότητες απομακρυσμένης διαχείρισης και εποπτείας μέσω διαδικτύου. Η παρακολούθηση, ο έλεγχος και η επεξεργασία δεδομένων σε πραγματικό χρόνο καθιστούν το σύστημα ευέλικτο, αυτοματοποιημένο και προσαρμοσμένο στις σύγχρονες ανάγκες του αγροτικού κλάδου.

Η δομή του κεφαλαίου διακρίνεται σε δύο βασικές υποενότητες:

- Πρώτον, την **παρουσίαση των βασικών λειτουργιών της κεντρικής μονάδας**, μέσα από το μενού επιλογών που προβάλλεται στην οθόνη LCD και υποστηρίζεται από τοπικά πλήκτρα.
- Δεύτερον, την **ανάλυση της υποδομής για τον απομακρυσμένο έλεγχο**, με έμφαση στη δημιουργία της HTML διεπαφής, την ενσωμάτωσή της στο σύστημα και τη σύνδεση με το cloud (Firebase), εξασφαλίζοντας επικοινωνία με το ESP32.

5.2 Λειτουργίες Κεντρικής Μονάδας

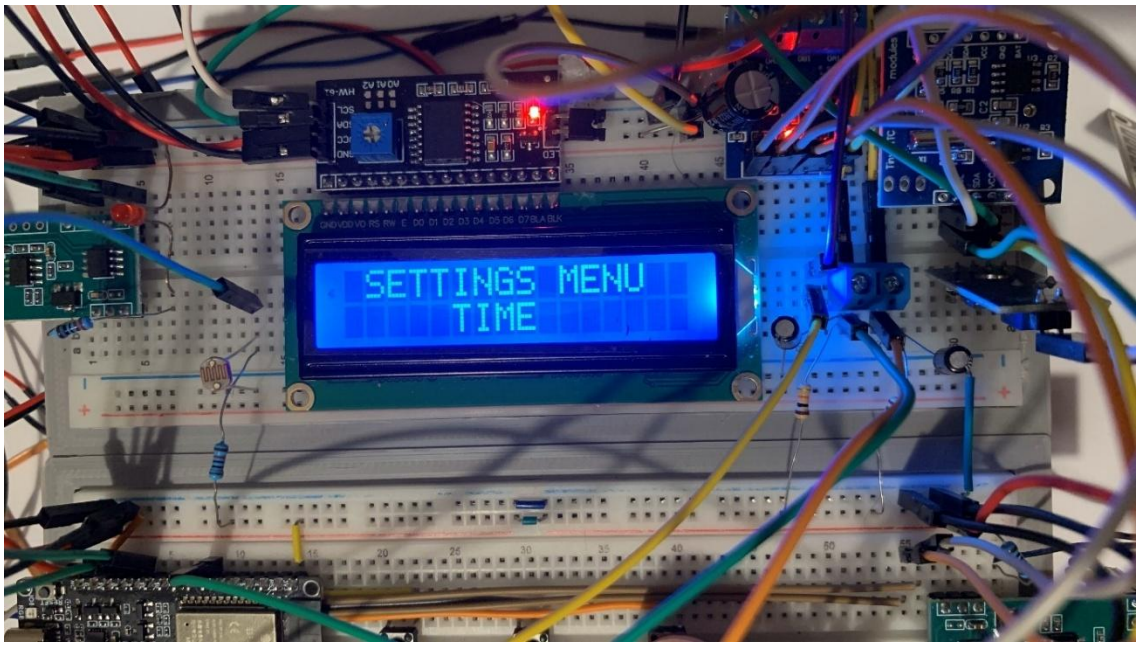
Η κεντρική μονάδα του αυτόματου θαλάμου αποτελεί το βασικό σημείο ελέγχου για τον χρήστη, προσφέροντας ένα απλό και φιλικό μενού πλοήγησης μέσω οθόνης LCD και πλήκτρων χειρισμού. Μέσα από τη συγκεκριμένη διεπαφή, ο χρήστης μπορεί να επιτηρεί βασικά δεδομένα, να παραμετροποιεί κρίσιμες λειτουργίες του συστήματος, αλλά και να ενεργοποιεί χειροκίνητα βασικούς μηχανισμούς, όπως η πόρτα και η ταΐστρα. Το μενού έχει σχεδιαστεί με λογική ιεράρχηση, ώστε κάθε κατηγορία να περιλαμβάνει υπομενού με στοχευμένες επιλογές.



Εικόνα 85: Στιγμιότυπο Αρχικής Οθόνης

Ακολουθεί αναλυτική παρουσίαση των βασικών λειτουργιών της κεντρικής μονάδας:

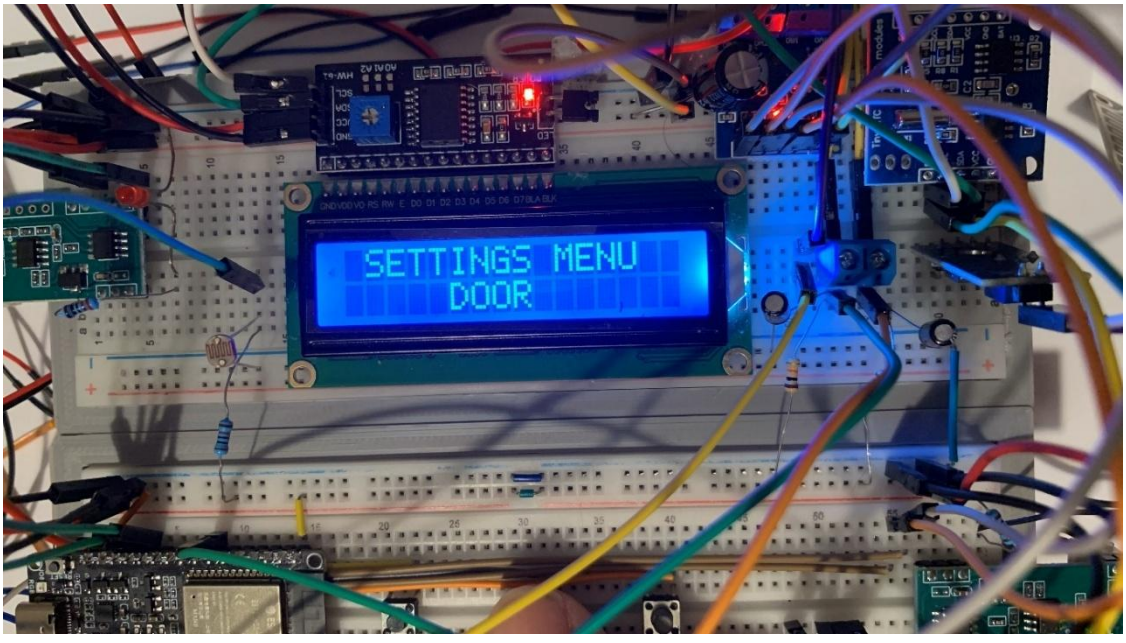
5.2.1 Time



Εικόνα 86: Menu Ρυθμίσεων Ώρας

Η επιλογή **Time** εμφανίζει την τρέχουσα ώρα και ημερομηνία, όπως αυτή διαβάζεται από το RTC (Real Time Clock) του συστήματος. Η ακριβής καταγραφή της ώρας είναι κρίσιμη, καθώς αποτελεί βάση για τις αυτοματοποιημένες ενέργειες, όπως το άνοιγμα/κλείσιμο της πόρτας και η ταΐστρα. Η ρύθμιση της ώρας γίνεται μία φορά κατά την αρχική παραμετροποίηση ή έπειτα από αλλαγή από θερινή σε χειμερινή.

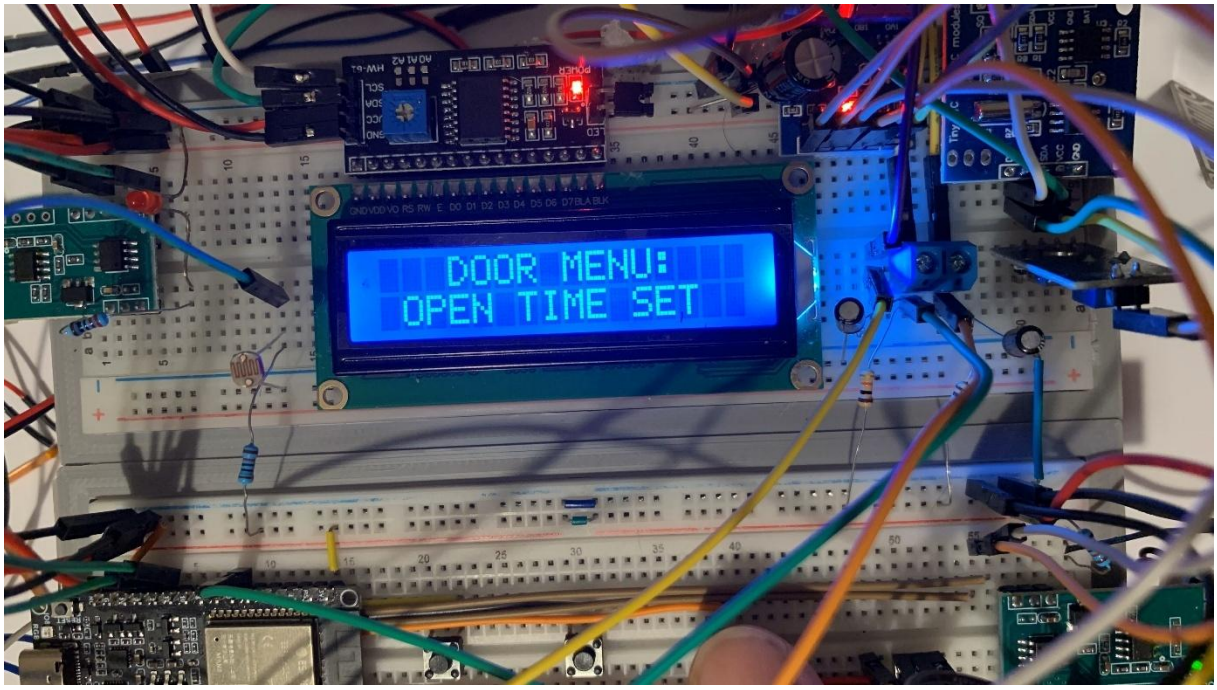
5.2.2 Door



Εικόνα 87: Menu Ρυθμίσεων Πόρτας

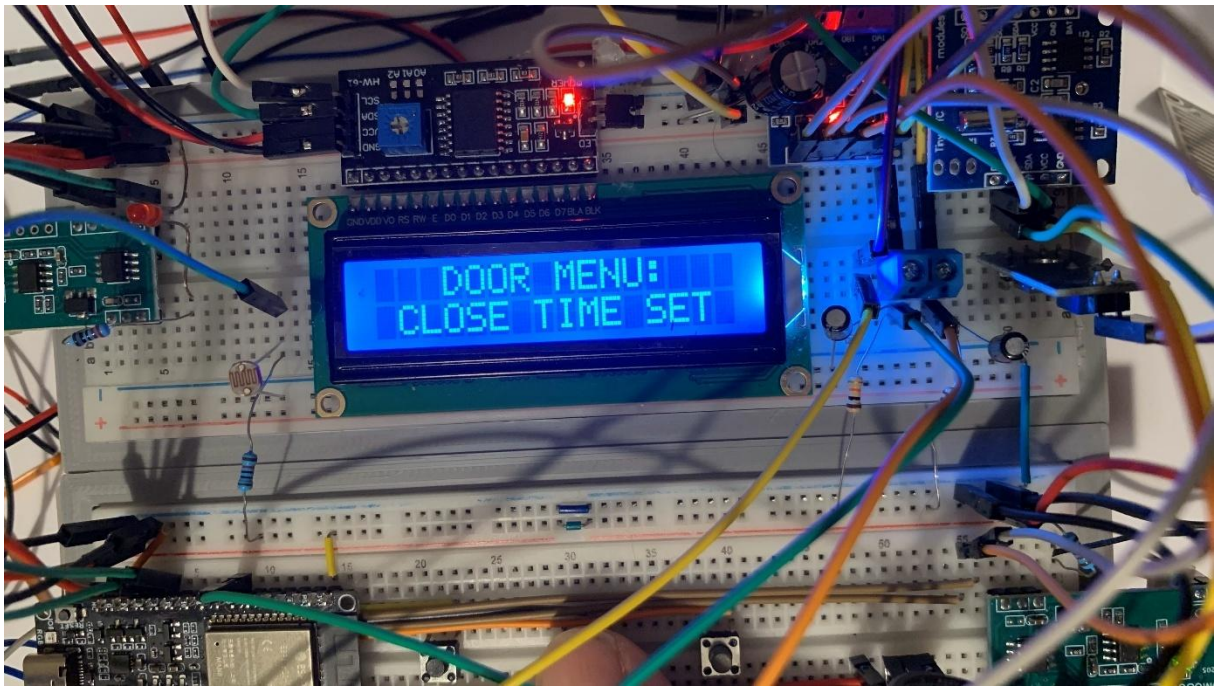
Η λειτουργία **Door** αφορά την αυτόματη και χειροκίνητη διαχείριση της πόρτας του θαλάμου. Περιλαμβάνει τρία υπομενού:

- **Open Time Set**: Επιτρέπει στον χρήστη να καθορίσει την ώρα που θα ανοίγει αυτόματα η πόρτα το πρωί.



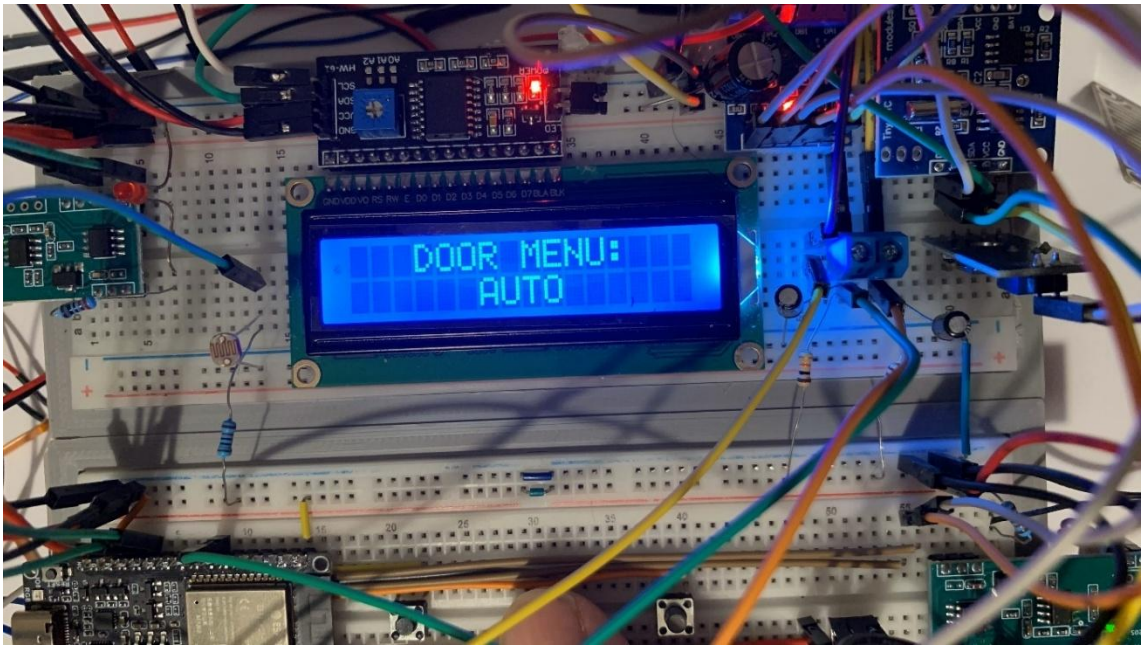
Εικόνα 88: Menu Ρυθμίσεων Ώρας Που θα ανοίξει η Πόρτα

- **Close Time Set**: Καθορίζει την ώρα αυτόματου κλεισίματος της πόρτας το βράδυ.



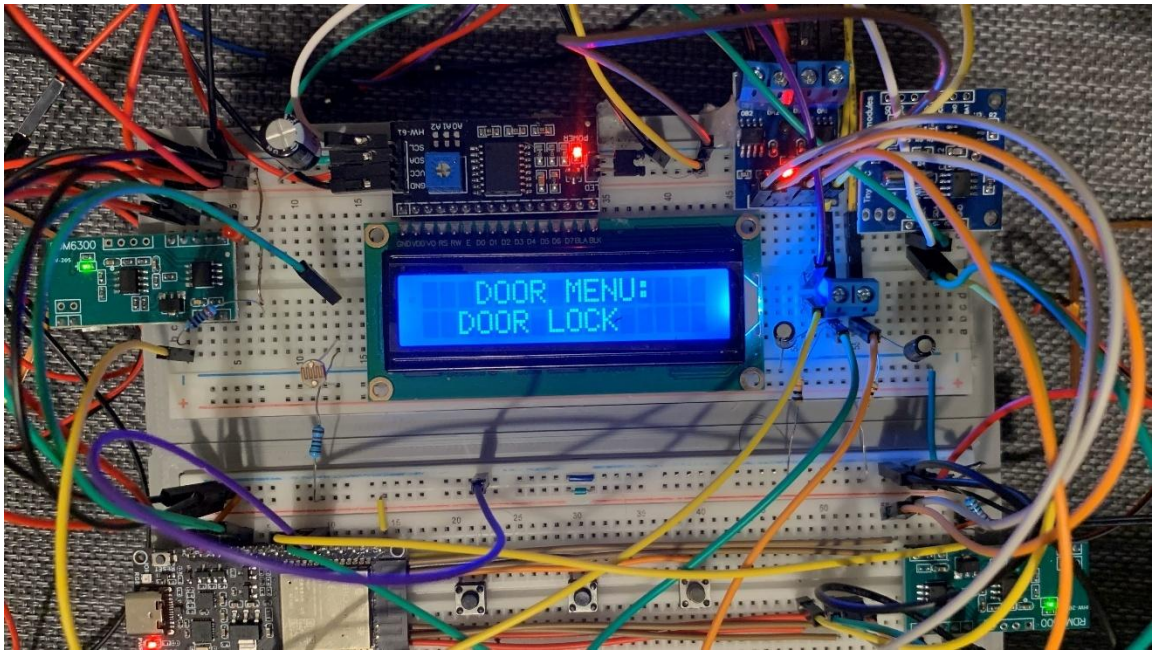
Εικόνα 89: Menu Ρυθμίσεων Ώρας Που θα Κλείσει η Πόρτα

- **Auto Door (ON/OFF)**: Ενεργοποιεί ή απενεργοποιεί τη λειτουργία αυτόματης πόρτας. Αν είναι ενεργοποιημένη, η πόρτα ανοίγει ή κλείνει ανάλογα την ανάγνωση του αισθητήρα φωτός ή χειροκίνητα από τον χρήστη.



Εικόνα 90: Menu Ρυθμίσεων Αυτόματης Λειτουργίας Πόρτας

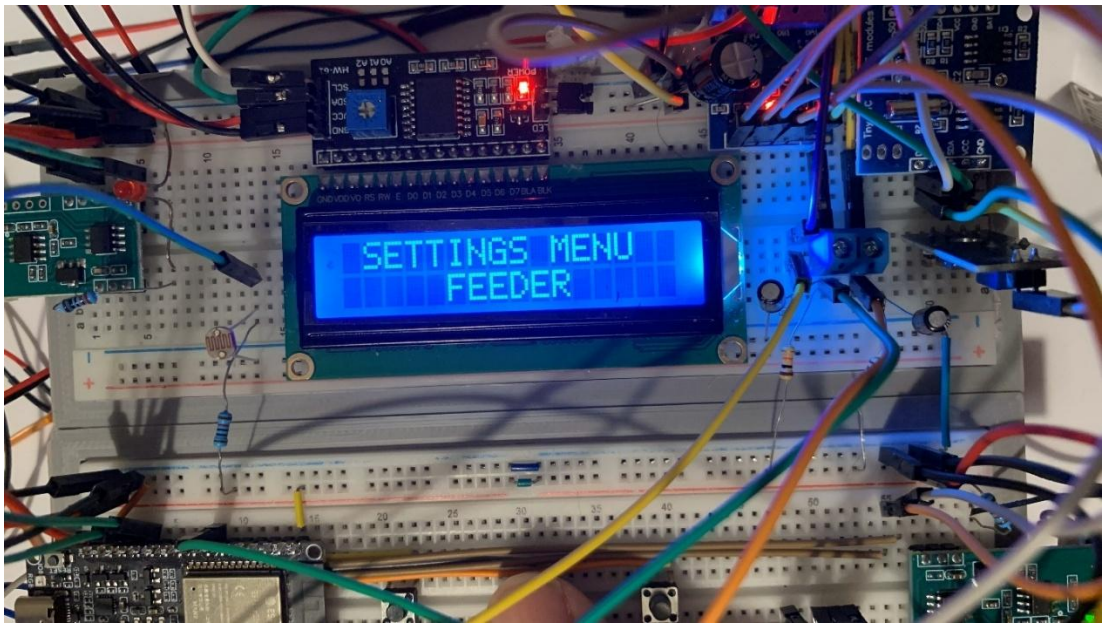
- **Door Lock:** υπαρχει επισης λειτουργια η οποια κλειδωνει την πορτα πατωντας το door lock. Η λειτουργια αυτή αναιρηται μονο παινοντας ξανα σε καποια από τις υπολοιπες ρυθμισεις της πορτας



Εικόνα 91: Menu DOOR LOCK

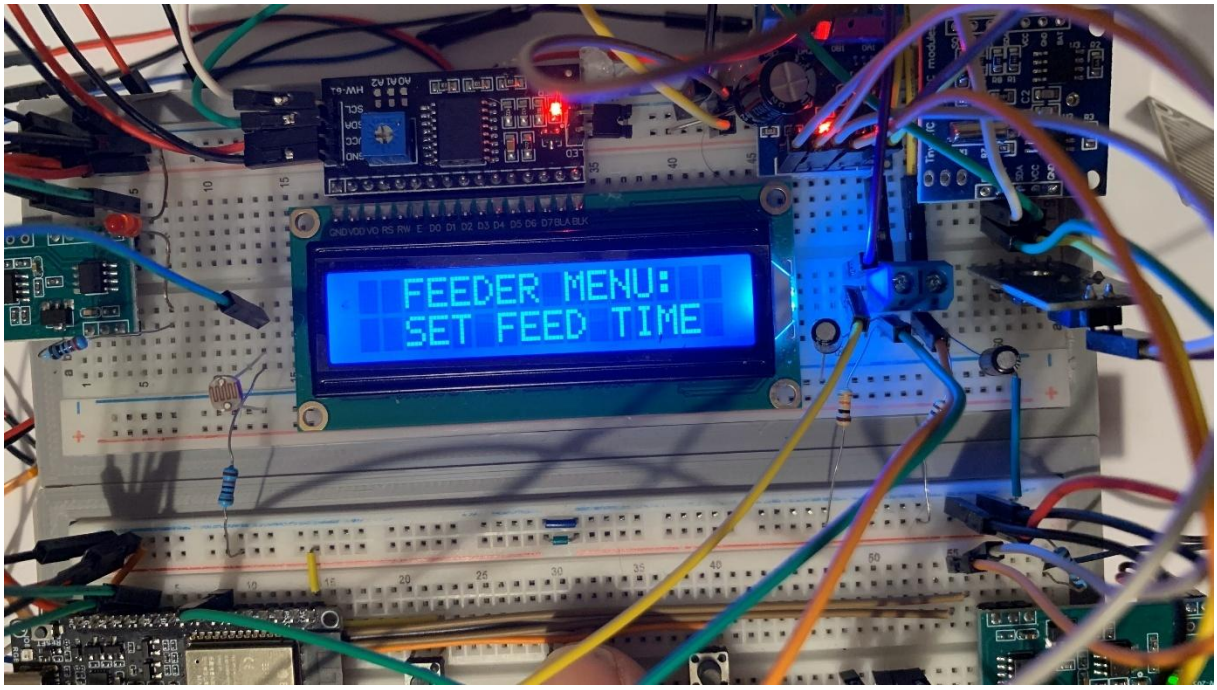
5.2.3 Feeder

Η επιλογή **Feeder** αφορά τη διαχείριση της αυτόματης ταΐστρας. Ο χρήστης μπορεί να καθορίσει παραμέτρους για το πότε και για πόσο χρόνο θα ενεργοποιείται η ταΐστρα:



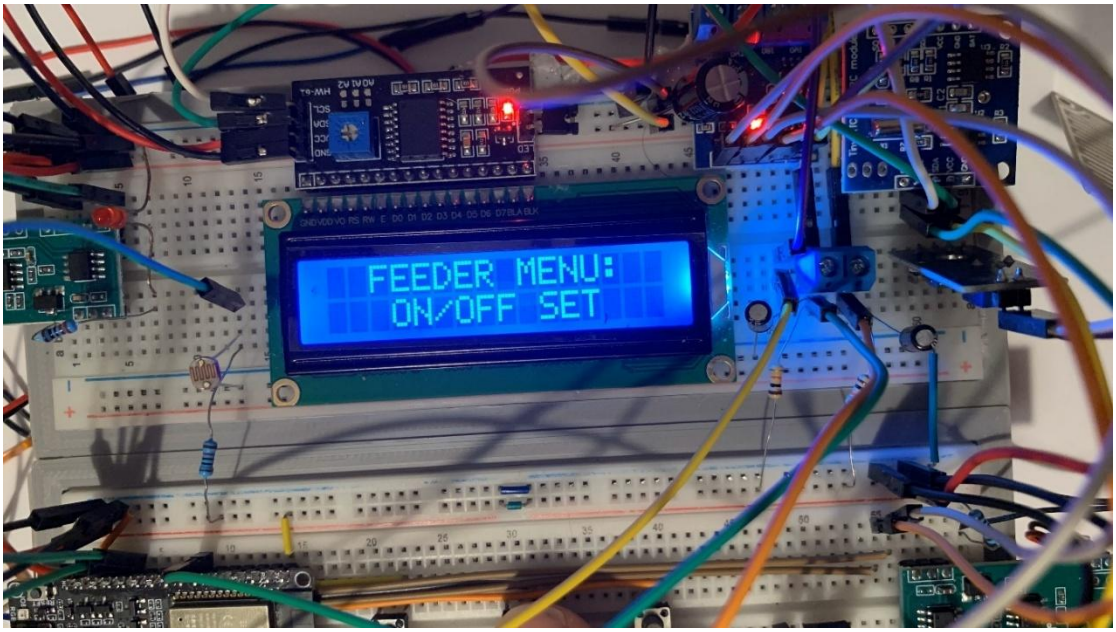
Εικόνα 92: Menu Ρυθμίσεων Ταΐστρας

- **Feed Time Set**: Ορίζει την ώρα κατά την οποία θα γίνεται η αυτόματη παροχή τροφής.



Εικόνα 93: Menu Ρυθμίσεων Ώρας Ταΐσματος

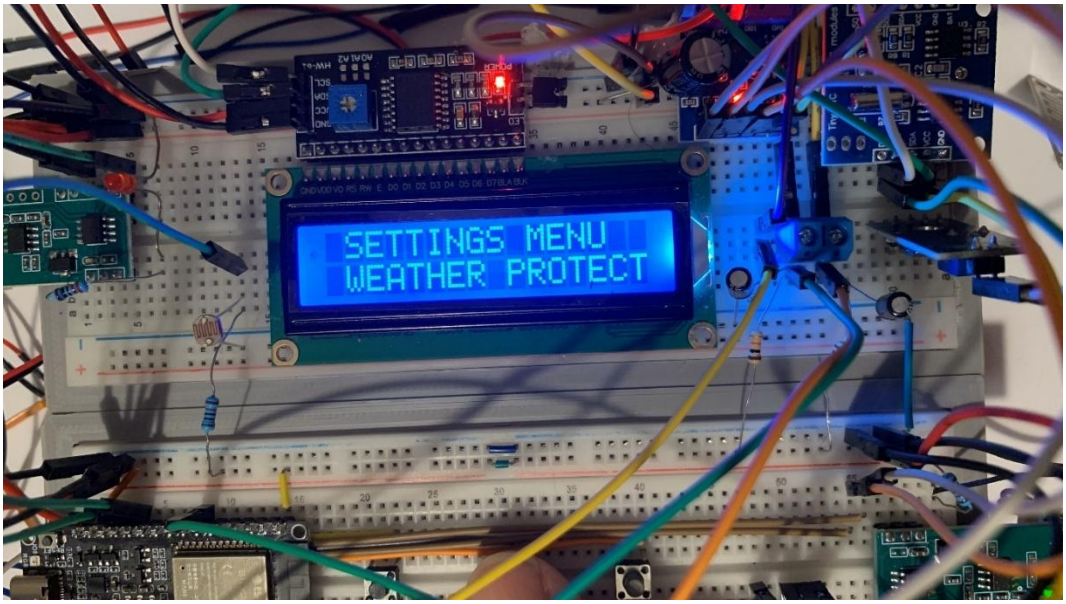
- **Duration**: Καθορίζει τη διάρκεια λειτουργίας του μοτέρ (δηλαδή για πόσα δευτερόλεπτα περιστρέφεται ο κοχλίας).
- **Feeder ON/OFF**: Ενεργοποιεί ή απενεργοποιεί συνολικά τη λειτουργία της αυτόματης ταΐστρας.



Εικόνα 94: Menu Ενεργοποίησης/Απενεργοποίησης Ταΐστρας

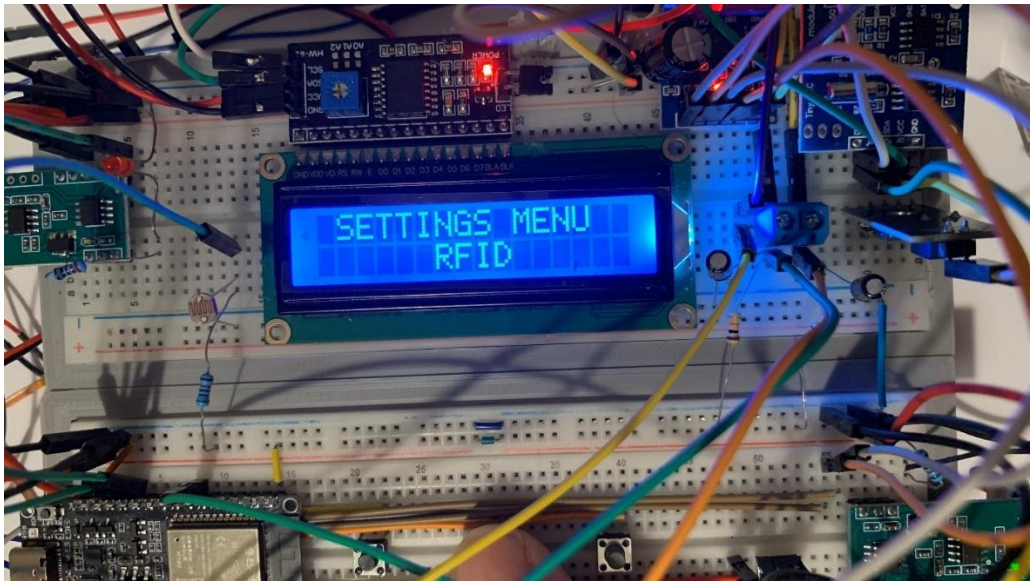
5.2.4 Weather Protect

Αυτή η επιλογή ενεργοποιεί μια λειτουργία προστασίας από δυσμενείς καιρικές συνθήκες. Όταν ανιχνευθεί βροχή από τον αντίστοιχο αισθητήρα ή πολύ χαμηλή θερμοκρασία, το σύστημα μπορεί να κρατήσει κλειστή την πόρτα ή να απενεργοποιήσει εξωτερικές λειτουργίες που θα μπορούσαν να επηρεαστούν από την υγρασία, διασφαλίζοντας έτσι την ασφάλεια του εξοπλισμού και των ζώων.



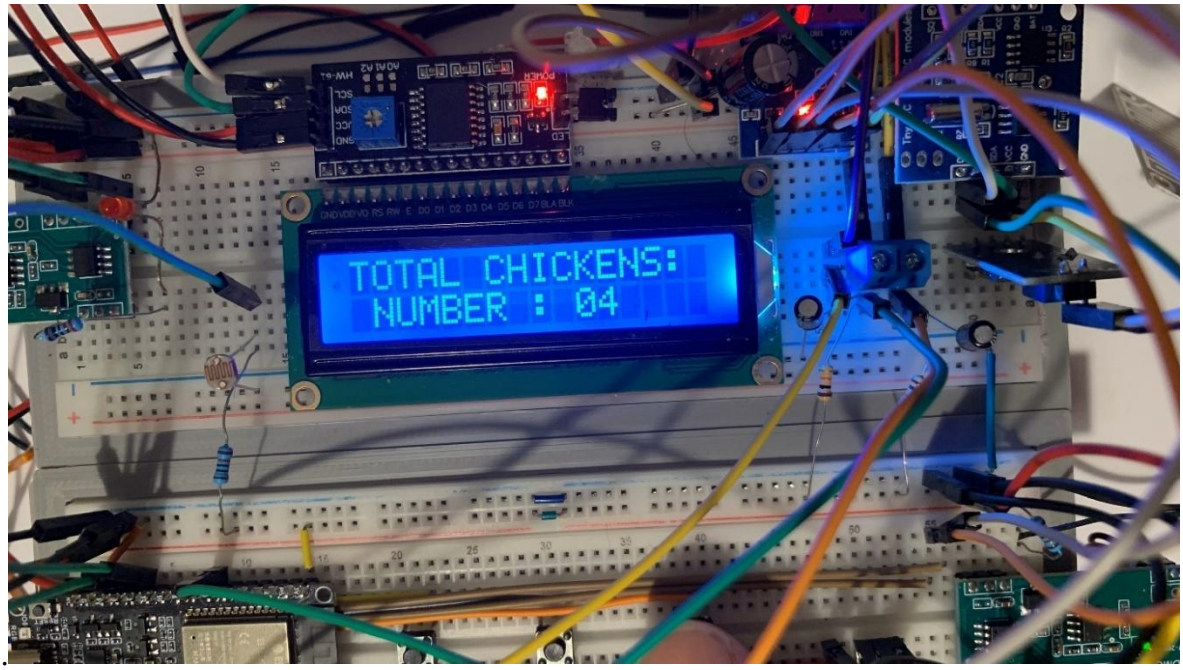
Εικόνα 95: Menu Ενεργοποίησης/Απενεργοποίησης Weather Protection

5.2.5 RFID



Εικόνα 96: Menu Ρυθμίσεων RFID

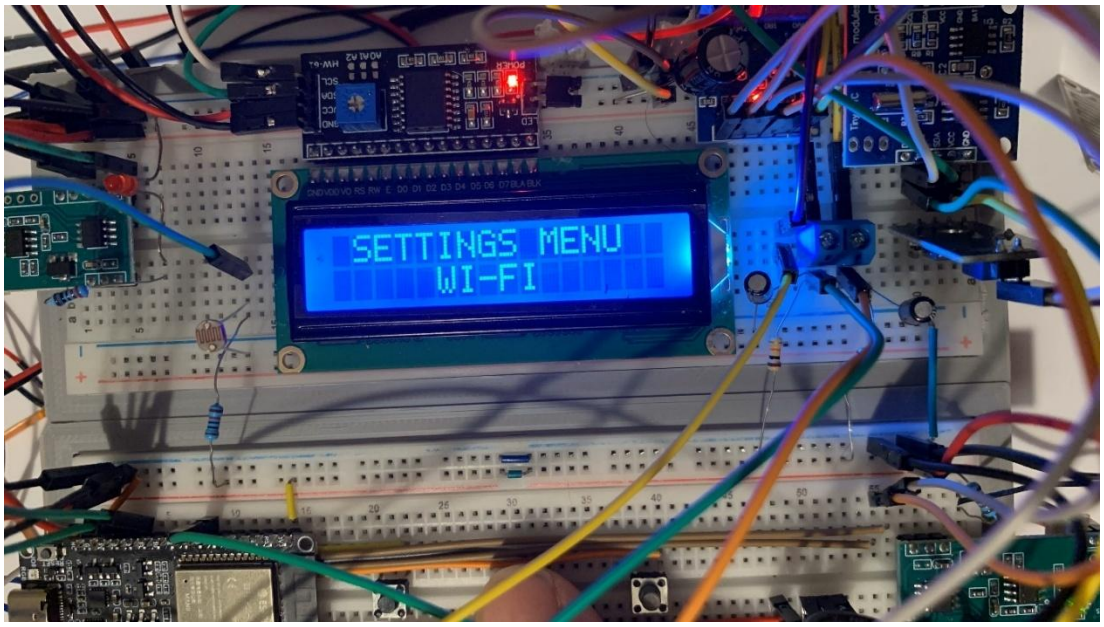
Το σύστημα ενσωματώνει τεχνολογία RFID για την αναγνώριση των ορνίθων μέσω tags/μπρελόκ. Η επιλογή **RFID** παρέχει πρόσβαση σε πληροφορίες που σχετίζονται με την αναγνώριση των ζώων και πόσα υπάρχουν στον θάλαμο. Ο συγκεκριμένος θάλαμος είναι σχεδιασμένος για μέχρι και πέντε κότες , παρόλα αυτά σε αυτό το μενού μπορούμε να ρυθμίσουμε αν έχουμε λιγότερες



Εικόνα 97: Menu Συνολικού Αριθμού Πτηνών Θαλάμου

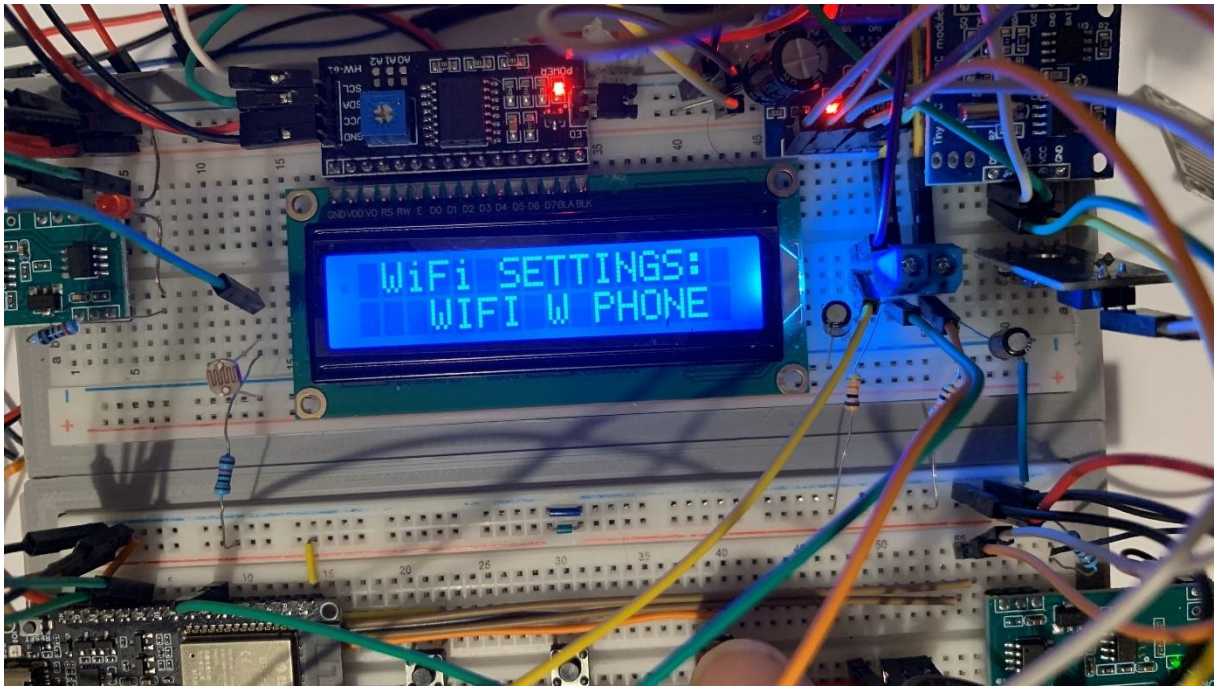
5.2.6 Wi-Fi

Το μενού **Wi-Fi** αφορά τις λειτουργίες σύνδεσης στο τοπικό δίκτυο ώστε να είναι δυνατός ο απομακρυσμένος έλεγχος. Περιλαμβάνει δύο βασικά υπομενού:



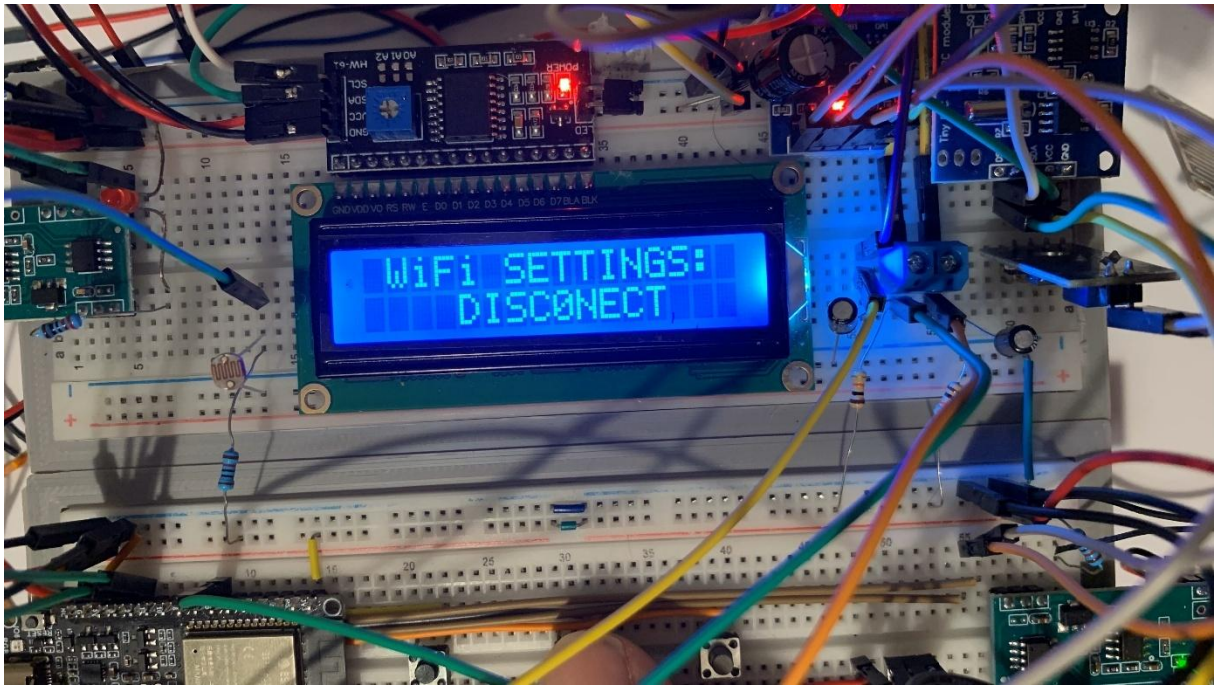
Εικόνα 98: Menu Ρυθμίσεων WiFi

- **Set with Phone**: Δίνει τη δυνατότητα στον χρήστη να ρυθμίσει τα στοιχεία δικτύου (SSID και κωδικό πρόσβασης) μέσω κινητού τηλεφώνου. Πρώτα χρειάζεται να συνδεθούμε στο wifi του esp32.



Εικόνα 99: Menu Ρυθμίσεων WiFi μέσω Κινητού Τηλεφώνου

- **Disconnect**: Διαγράφει τις αποθηκευμένες ρυθμίσεις Wi-Fi, αποσυνδέεται και επαναφέρει το σύστημα σε κατάσταση αναμονής για νέα σύνδεση.



Εικόνα 100: Menu Αποσύνδεσης Και Διαγραφής WiFi

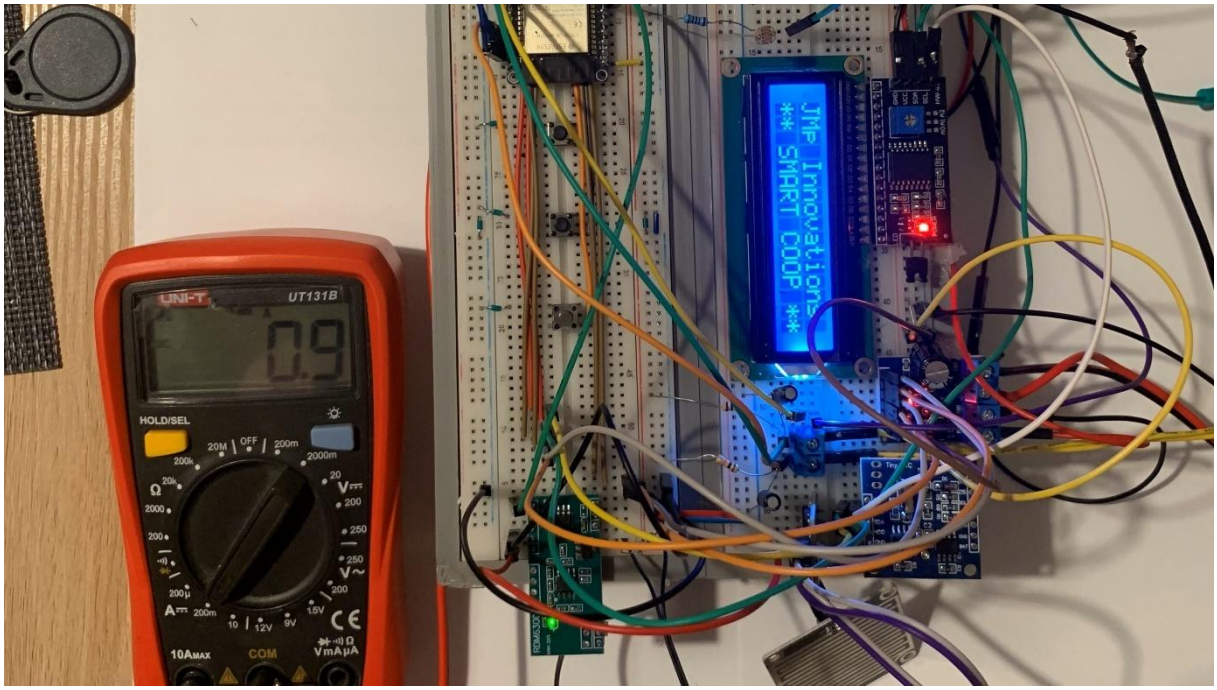
5.2.7 Μετρήσεις

Η αξιολόγηση της ενεργειακής συμπεριφοράς ενός αυτόνομου συστήματος αποτελεί βασικό στάδιο τόσο για τη βελτιστοποίηση της απόδοσής του όσο και για τη μελλοντική του επέκταση σε συνθήκες που δεν εξασφαλίζεται σταθερή ηλεκτρική τροφοδοσία. Στο πλαίσιο της παρούσας εφαρμογής, πραγματοποιήθηκαν μετρήσεις ηλεκτρικής κατανάλωσης για να καθοριστεί με ακρίβεια η ισχύς που απαιτείται σε κάθε φάση λειτουργίας του θαλάμου. Οι μετρήσεις αυτές επιτρέπουν την καταγραφή της ενεργειακής συμπεριφοράς των επιμέρους υποσυστημάτων, την εκτίμηση της διάρκειας λειτουργίας υπό συγκεκριμένη παροχή ενέργειας (π.χ. με μπαταρίες ή φωτοβολταϊκά), καθώς και την εντολή κρίσιμων καταναλωτών με σκοπό τη βελτίωση της αποδοτικότητας.

Οι καταγραφές πραγματοποιήθηκαν υπό ρεαλιστικές συνθήκες, με το σύστημα να λειτουργεί πλήρως και με ενεργοποιημένα όλα τα κρίσιμα υποσυστήματα, όπως οι αισθητήρες, τα μοτέρ, η οθόνη και η μονάδα RFID. Η μέθοδος που χρησιμοποιήθηκε περιλάμβανε την παρακολούθηση της κατανάλωσης ρεύματος μέσω αμπερομέτρου και μετρητή ισχύος σε διαδοχικές καταστάσεις: ηρεμίας, ενεργοποίησης εκτελεστικών μηχανισμών (π.χ. πόρτας, ταΐστρας) και συνεχούς λειτουργίας. Έμφαση δόθηκε στην αποτύπωση των στιγμιαίων αιχμών ισχύος που προκύπτουν από κινητήρες ή άλλες μεταβαλλόμενες φορτίσεις, καθώς και στην καταγραφή της μέσης ημερήσιας κατανάλωσης σε τυπικό κύκλο λειτουργίας.

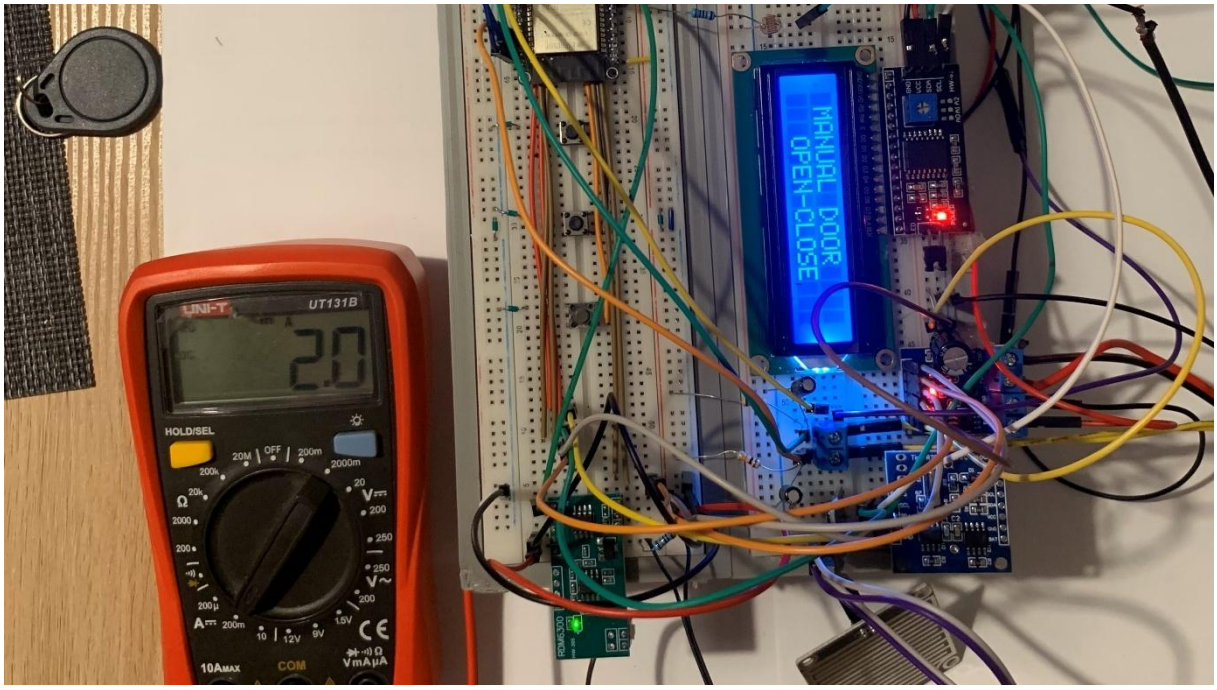
Συνοπτικά, καταγράφηκαν τα παρακάτω:

- **Κατανάλωση σε κατάσταση ηρεμίας (idle):** ~0,9 mA στα 5V
  Περιλαμβάνει μόνο τη βασική λειτουργία του ESP32, την οθόνη LCD και τους βασικούς αισθητήρες (θερμοκρασίας, παρουσίας).



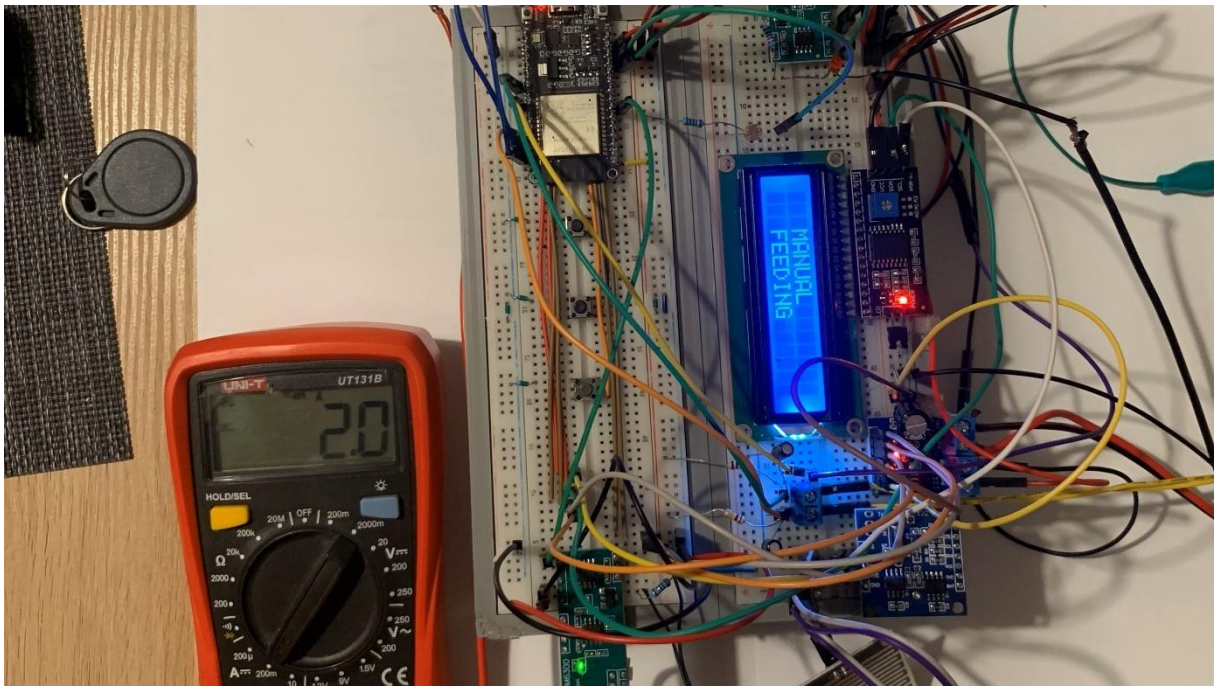
Εικόνα 101: Μέτρηση σε φάση Ηρεμίας

- **Κατανάλωση κατά την ενεργοποίηση της πόρτας:** ~2 mA στα 5V για ~50 δευτερόλεπτα



Εικόνα 102: Μέτρηση Κατά την διάρκεια Ανοίγματος της Πόρτας

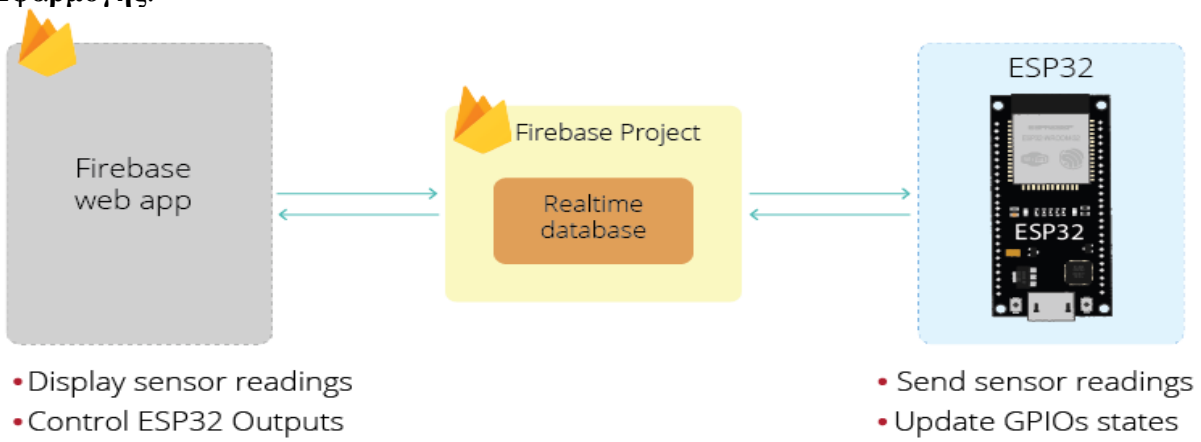
- **Κατανάλωση κατά τη λειτουργία της ταΐστρας:** ~2 mA στα 5V για διάρκεια ~4–5 δευτερόλεπτα
  Η κατανάλωση αφορά το μοτέρ και τον οδηγό περιστροφής του κοχλία.



Εικόνα 103: Μέτρηση Κατά την Διάρκεια λειτουργίας της Ταΐστρας

5.3 Υποσύστημα Απομακρυσμένου Ελέγχου

Η δυνατότητα απομακρυσμένης παρακολούθησης και ελέγχου του αυτόματου θαλάμου αποτελεί βασικό χαρακτηριστικό ενός σύγχρονου, έξυπνου συστήματος. Στο παρόν υποσύστημα, ο μικροελεγκτής ESP32 επικοινωνεί ασύρματα με την πλατφόρμα **Firebase** της Google, μέσω Wi-Fi. Η πλατφόρμα αυτή λειτουργεί ως ενδιάμεσος κόμβος, αποθηκεύοντας και διαχειριζόμενη σε πραγματικό χρόνο όλα τα δεδομένα και τις ρυθμίσεις του θαλάμου, επιτρέποντας στο χρήστη να αλληλεπιδρά μαζί τους μέσω μιας **ειδικά σχεδιασμένης Web Εφαρμογής**.



Εικόνα 104: Firebase Communication With Web App Πηγή: [firebase esp32 web app - Αναζήτηση Google]

Το υποσύστημα αυτό αναλύεται σε τρία βασικά τμήματα:

5.3.1 Επικοινωνία ESP32 με Firebase

Ο μικροελεγκτής **ESP32** διαθέτει ενσωματωμένη δυνατότητα Wi-Fi, κάτι που του επιτρέπει να συνδέεται απευθείας στο διαδίκτυο. Στο σύστημα της πτυχιακής, χρησιμοποιείται για την επικοινωνία με την υπηρεσία **Firebase Realtime Database**.

Για την επικοινωνία αυτή:

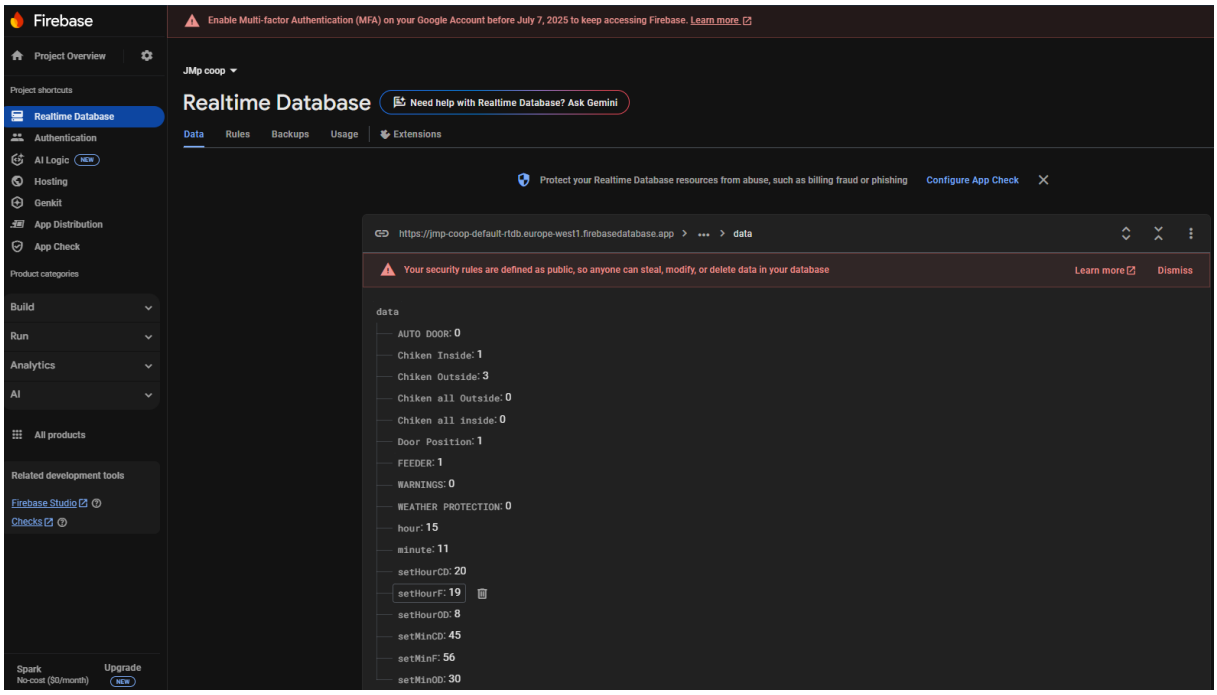
- Ενσωματώθηκαν ειδικές βιβλιοθήκες Firebase για Arduino, όπως η Firebase_ESP_Client, που διευκολύνουν την αρχικοποίηση και τον συγχρονισμό.
- Το ESP32 πραγματοποιεί **read και write εντολές** προς τη βάση δεδομένων.
- Τα δεδομένα όπως θερμοκρασία, κατάσταση πόρτας ή RFID αναγνωρίσεις αποστέλλονται στο cloud.
- Αντίστοιχα, το ESP32 "ακούει" για αλλαγές (π.χ. εντολή για άνοιγμα ταΐστρας) και εκτελεί τις αντίστοιχες ενέργειες.

Αυτή η ασύρματη επικοινωνία καθιστά δυνατή την απομακρυσμένη πρόσβαση στον θάλαμο, χωρίς την ανάγκη τοπικής παρουσίας του χρήστη.

```
#include <LCD-I2C.h>
#include <Wire.h>
#include <RTClib.h>
#include <EEPROM.h>
#include <WiFi.h>
#include <WiFiManager.h>
#include <esp_wifi.h>
#include <esp_wps.h>
#include <Firebase_ESP_Client.h>
#include <SoftwareSerial.h>

SoftwareSerial readerIn(0, -1);  // RX, TX pins για reader εισόδου
SoftwareSerial readerOut(2, -1);   // RX, TX pins για reader εξόδου

String allowedTags[5] = {
  "31004187AB",
  "0D002567A7",
  "1800793145",
  "05003860F7",
  "3800C5A41D"
 };
// Κατάσταση για κάθε tag (true = μέσα, false = έξω)
bool insideStatus[5] = {false, false, false, false, false};

int MaxCknNo=4;

int chknNO=4;
int in=0;
int out=0;

#define I2C_SDA 21
#define I2C_SCL 22
#define EEPROM_SIZE 512
#define SSID_ADDR 8
#define PASS_ADDR 9
#define MAX_SSID_LEN 32
#define MAX_PASS_LEN 64
#define WIFI_TIMEOUT_MS 10000
#define WPS_MODE WPS_TYPE_PBC
esp_wps_config_t wpsConfig = WPS_CONFIG_INIT_DEFAULT(WPS_MODE);

#define DATABASE_URL "https://jmp-coop-default-rtdb.europe-west1.firebasedatabase.app/"
#define API_KEY "AIzaSyDUMuRZCncdTnVhhVRHwDd_kjY2vJW6ST0"
#define USER_EMAIL "esp32@jmp.com"           // Εσύ το ορίζεις
#define USER_PASSWORD "12345678"

```

Εικόνα 105: Βιβλιοθήκη Firebase και APIS

5.3.2 Firebase Realtime Database



Εικόνα 106: Στιγμιότυπο από την οθόνη Real Time Database του Firebase

Το **Firebase Realtime Database** είναι μία cloud-based, NoSQL βάση που αποθηκεύει δεδομένα με μορφή **JSON**. Ο κύριος της ρόλος στο έργο είναι να λειτουργεί σαν ένας κοινός «χώρος συνάντησης» μεταξύ του ESP32 και του web περιβάλλοντος του χρήστη.

```
289       // Αντικείμενο JSON για αποστολή
290     FirebaseJson json;
291     json.set("hour", now.hour());
292     json.set("minute", now.minute());
293     json.set("AUTO DOOR", doorA);
294     json.set("setHourOD", setHourOD);
295     json.set("setMinOD", setMinOD);
296     json.set("setHourCD", setHourCD);
297     json.set("setMinCD", setMinCD);
298     json.set("FEEDER", feederST);
299     json.set("setHourF", setHourF);
300     json.set("setMinF", setMinF);
301     json.set("WEATHER PROTECTION", WthrPR);
302     json.set("WARNINGS", wrn);
303     json.set("Chiken Inside", insideCount);
304     json.set("Chiken Outside", outsideCount);
305     json.set("Chiken all inside", in);
306     json.set("Chiken all Outside", out);
307     json.set("Door Position", doorST);
308
309     // Στείλε τα δεδομένα στο path "/esp32/data"
310     if (Firebase.RTDB.setJSON(&fbdo, "/esp32/data", &json)) {
311       Serial.println("Data sent to Firebase");
312     } else {keepConnected();
313       Serial.print("Firebase error: ");
314       Serial.println(fbdo.errorReason());
315     }
316   }
```

Εικόνα 107: Πακέτα Json από ESP για Firebase

Κύρια χαρακτηριστικά:

- Κάθε φορά που ο μικροελεγκτής ενημερώνει μία τιμή, αυτή ενημερώνεται **σε πραγματικό χρόνο**.
- Ο χρήστης από το Web App μπορεί να αλλάξει παραμέτρους (όπως ώρα ταΐσματος), οι οποίες καταγράφονται άμεσα και διαβάζονται από το ESP32.
- Υποστηρίζει **κανόνες ασφαλείας και αυθεντικοποίησης**, προστατεύοντας την ακεραιότητα των δεδομένων.

Η δομή της βάσης μπορεί να περιλαμβάνει κόμβους όπως /doorST, /feederST, με ξεκάθαρη ιεραρχία για εύκολη αναζήτηση και τροποποίηση.

Επίσης κάθε φορά που ο χρήστης ενημερώνει μια τιμή από την εφαρμογή πάλι μέσο του firebase ενημερώνεται και η τιμή στο φυσικό σύστημα

```
void receiveIOT(){
  if (millis() - lastCommandTime > 2000) {
    lastCommandTime = millis();

    if (Firebase.RTDB.getJSON(&fbdo, "/esp32/commands")) {
      FirebaseJson &json = fbdo.jsonObject();
      FirebaseJsonData cmdType;
      if (json.get(cmdType, "type")) {
        if (cmdType.stringValue == "manual_feed") {
          Serial.println("Received manual_feed command!");
          FEED();
          clearCommand(); }
        else if (cmdType.stringValue == "toggle_door") {
          Serial.println("Received manual open/close Door command!");
          doorOpenClose();
          clearCommand(); }
        else if (cmdType.stringValue == "update_settings") {
          FirebaseJsonData weatherVal, feederVal, autoDoorVal;
          FirebaseJsonData temp;
                    // Weather protection
          if (json.get(weatherVal, "weather")) {
            int remoteWthrPR = weatherVal.boolValue ? 1 : 0;
            if (WthrPR != remoteWthrPR) {
              WthrPR = remoteWthrPR;
              Serial.println("Weather protection updated remotely: "); }}
                    // Feeder
          if (json.get(feederVal, "feeder")) {
            int remoteFeeder = feederVal.boolValue ? 1 : 0;
            if (feederST != remoteFeeder) {
              feederST = remoteFeeder;
              Serial.println("feeder on/off updated remotely: "); }}
                    // Auto Door
          if (json.get(autoDoorVal, "autoDoor")) {
            int remoteDoor = autoDoorVal.boolValue ? 1 : 0;
            if (doorA != remoteDoor) {
              doorA = remoteDoor;
              Serial.print("Auto Door updated updated remotely: ");}}
                              // Feed Time
          if (json.get(temp, "feederHour")) setHourF = temp.intValue;
          if (json.get(temp, "feederMin"))  setMinF  = temp.intValue;
                            // Open Door Time
          if (json.get(temp, "openHour"))   setHourOD = temp.intValue;
          if (json.get(temp, "openMin"))    setMinOD  = temp.intValue;
                          // Close Door Time
          if (json.get(temp, "closeHour"))  setHourCD = temp.intValue;
          if (json.get(temp, "closeMin"))   setMinCD  = temp.intValue;
```

Εικόνα 108: Δεδομένα λήψης από Web App σε Firebase και ESP32

5.3.3 5Ανάπτυξη Web Εφαρμογής

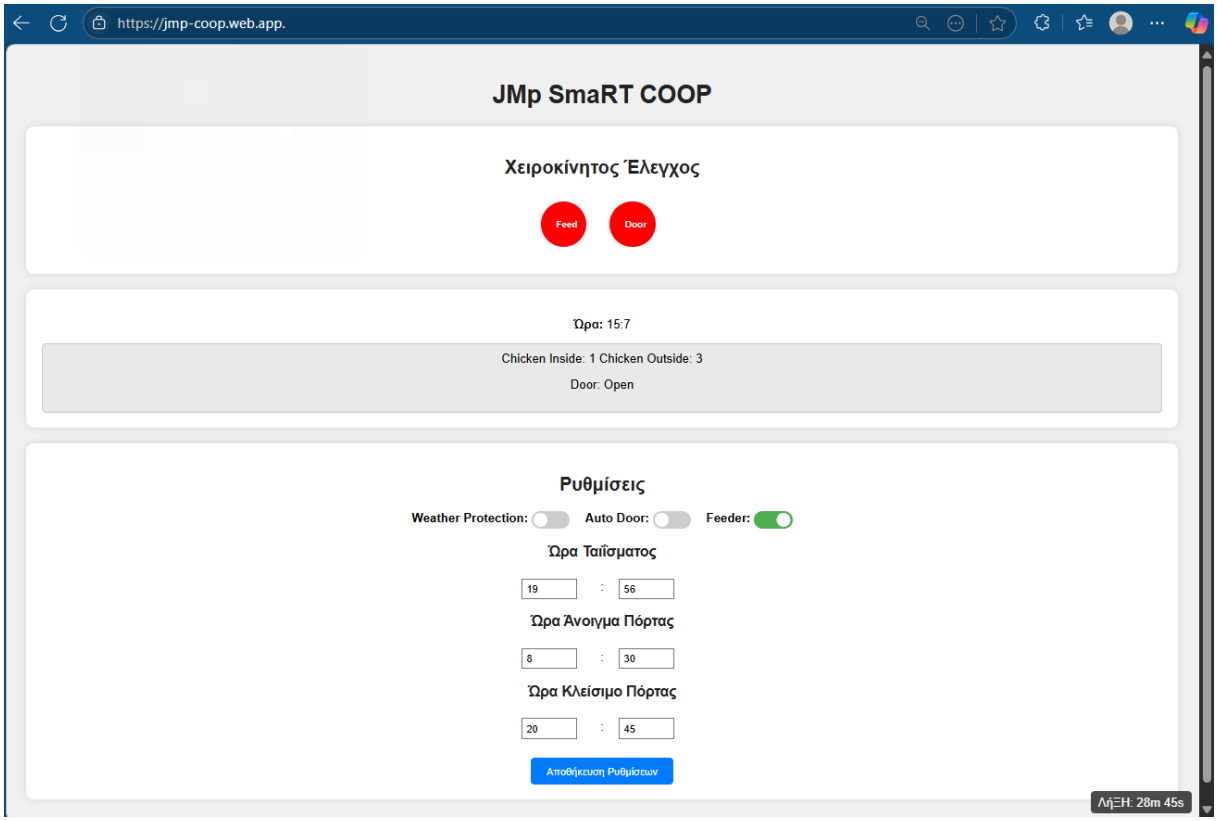
Για τον απομακρυσμένο έλεγχο, σχεδιάστηκε μια **απλή αλλά λειτουργική Web Εφαρμογή**. Η Web Εφαρμογή που δημιουργήθηκε αποτελεί το κύριο μέσο αλληλεπίδρασης του χρήστη με το σύστημα. Είναι πρόσβαση από οποιονδήποτε φυλλομετρητή, είτε από υπολογιστή είτε από κινητή συσκευή, χωρίς την ανάγκη εγκατάστασης επιπλέον εφαρμογών. Αυτή αναπτύχθηκε με χρήση:

- **HTML/CSS/JavaScript** για τη διαμόρφωση της διεπαφής χρήστη (UI),
- και του **Firebase Web SDK**, για την αλληλεπίδραση με τη Realtime Database.

Λειτουργίες της εφαρμογής:

- Εμφάνιση δεδομένων σε πραγματικό χρόνο (όπως κατάσταση πορτών, χρόνοι ταΐσματος).
- Δυνατότητα ρύθμισης παραμέτρων (π.χ. ώρα ανοίγματος πόρτας, ώρα ταΐσματος).
- Ενεργοποίηση/απενεργοποίηση συστημάτων (όπως αυτόματη πόρτα, αυτόματο τάισμα).
- Εμφάνιση ειδοποιήσεων (όπως χαμηλή στάθμη φαγητού)

Η εφαρμογή μπορεί να φορτωθεί από οποιαδήποτε συσκευή με φυλλομετρητή, προσφέροντας **ευκολία χρήσης**, ακόμα και μέσω κινητού ή Ταμπλετ.



Εικόνα 109: Στιγμιότυπο της οθόνης Από το Web-App

5.4 Επίλογος Κεφαλαίου

Το πέμπτο κεφάλαιο αποτέλεσε την ολοκλήρωση της τεχνικής περιγραφής του αυτόματου θαλάμου, παρουσιάζοντας αναλυτικά τον τρόπο με τον οποίο το σύστημα λειτουργεί τόσο τοπικά όσο και απομακρυσμένα. Μέσα από τη δομημένη παρουσίαση του μενού λειτουργιών της κεντρικής μονάδας, αναδείχθηκε η ευελιξία, η αυτονομία και η επεκτασιμότητα του συστήματος.

Η ενσωμάτωση του Firebase σε συνδυασμό με τον μικροελεγκτή ESP32 αποδείχθηκε καθοριστικής σημασίας για την επιτυχία του έργου. Χάρη στη σύνδεση με την **Realtime Database** και την ανάπτυξη της **web εφαρμογής**, επιτεύχθηκε πλήρης απομακρυσμένος έλεγχος και παρακολούθηση των κρίσιμων παραμέτρων του θαλάμου. Η δυνατότητα άμεσης ενημέρωσης των τιμών σε πραγματικό χρόνο, σε συνδυασμό με τη λειτουργικότητα της κεντρικής μονάδας, διασφαλίζει την αδιάλειπτη και αποτελεσματική λειτουργία του συστήματος, με ελάχιστη ανάγκη για ανθρώπινη παρέμβαση.

Συνολικά, το υποσύστημα απομακρυσμένου ελέγχου καθιστά τον αυτόματο θάλαμο μια πλήρως έξυπνη εφαρμογή, ικανή να λειτουργεί αυτόνομα, να προσαρμόζεται δυναμικά στις συνθήκες και να ενημερώνει τον χρήστη από οπουδήποτε και αν βρίσκεται. Η αρμονική συνύπαρξη φυσικής και ψηφιακής διεπαφής αποδεικνύει τη λειτουργική πληρότητα και την τεχνολογική ωριμότητα της παρούσας κατασκευής.

Κεφάλαιο 6ο: Συμπεράσματα, Βελτιώσεις και Επεκτάσεις

Η παρούσα πτυχιακή εργασία είχε ως στόχο τον σχεδιασμό και την υλοποίηση ενός αυτόματοποιημενου θαλάμου για όρνιθες, το οποίο να συνδυάζει τεχνολογίες αυτοματισμού, αισθητήρων και απομακρυσμένου ελέγχου μέσω διαδικτύου. Μέσα από τη μελέτη, την κατασκευή και τη λειτουργική εφαρμογή του συστήματος, κατέστη σαφές ότι ακόμη και σε μικρής κλίμακας εφαρμογές αγροτικής παραγωγής, η χρήση τεχνολογικών λύσεων μπορεί να επιφέρει ουσιαστική βελτίωση τόσο στην απόδοση όσο και στην ευκολία της καθημερινής διαχείρισης.

Το σύστημα πέτυχε την πλήρη αυτοματοποίηση βασικών λειτουργιών όπως το άνοιγμα και το κλείσιμο της πόρτας, η παροχή τροφής, η αναγνώριση των ζώων μέσω RFID και η προστασία από καιρικές συνθήκες. Παράλληλα, ο χρήστης έχει τη δυνατότητα απομακρυσμένου ελέγχου μέσω μιας απλής αλλά λειτουργικής διαδικτυακής εφαρμογής, που επικοινωνεί σε πραγματικό χρόνο με το ESP32 μέσω της πλατφόρμας Firebase. Η χρήση φθηνών αλλά αξιόπιστων υλικών, σε συνδυασμό με τρισδιάστατα εκτυπωμένα εξαρτήματα, αποδεικνύει ότι η κατασκευή τέτοιων συστημάτων μπορεί να είναι προσιτή και εύκολα επαναλήψιμη.

Προτάσεις Βελτίωσης

Ωστόσο, όπως κάθε τεχνικό έργο, έτσι και αυτό προσφέρει σημαντικά περιθώρια για εξέλιξη. Μελλοντικά, το σύστημα θα μπορούσε να εμπλουτιστεί με περισσότερες δυνατότητες και να αποκτήσει μεγαλύτερη αυτονομία και ευφυΐα. Η ενσωμάτωση κάμερας IP για οπτική παρακολούθηση, η προσθήκη 4G MODEM για δικτύωση ακόμα και χωρίς υπάρχων wifi, χρήση UHF Rfid για μεγαλύτερη κάλυψη και αξιοπιστία δυνατότητα προγραμματισμού μέσω OTA και η χρήση τεχνητής νοημοσύνης για την πρόβλεψη αναγκών βάσει ιστορικών δεδομένων είναι μόνο μερικές από τις κατευθύνσεις στις οποίες μπορεί να κινηθεί η εξέλιξη του έργου. Επιπλέον, η ανάπτυξη ενός πιο σύνθετου web περιβάλλοντος με ειδοποιήσεις, ιστορικά στοιχεία και πολλαπλά επίπεδα πρόσβασης, θα αύξανε σημαντικά τη λειτουργικότητα και την προσαρμοστικότητα του συστήματος.

Συνολικά, το έργο αυτό αποτέλεσε μια εξαιρετικά χρήσιμη εμπειρία, καθώς συνδύασε θεωρητική γνώση και πρακτική εφαρμογή, δίνοντας την ευκαιρία να εφαρμοστούν σύγχρονες τεχνολογίες στην πράξη και να δημιουργηθεί μια ολοκληρωμένη λύση με πραγματική χρησιμότητα και προοπτικές εξέλιξης.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] M. Appleby, B. Mench, and J. Hughes, *Poultry Behaviour and Welfare*, CABI Publishing, 2004.

[2] FAO, "Poultry Production Systems," Food and Agriculture Organization of the United Nations. [Online]. Available: https://www.fao.org/poultry-production/en/

[3] The Poultry Site, "Chicken Welfare and Housing," [Online]. Available: https://www.thepoultrysite.com/articles/categories/husbandry-welfare

[4] University of Kentucky Cooperative Extension, "Raising Chickens," [Online]. Available: https://afs.ca.uky.edu/poultry/raising-chickens

[5] A. Wolfert, L. Ge, C. Verdouw, and M.-J. Bogaardt, "Big Data in Smart Farming – A review," *Agricultural Systems*, vol. 153, pp. 69–80, 2017.

[6] Backyard & Small Poultry Flock Management Series: Feeding the Laying Hen

https://www.aces.edu/blog/topics/farming/backyard-small-poultry-flock-management-series-feeding-the-laying-hen/

[7] How Much Water Do Chickens Need? Expert Tips for Keeping Your Flock Hydrated

https://www.dineachook.com.au/blog/how-much-water-do-chickens-need/

[8] K. Ashton, "That 'Internet of Things' Thing," *RFID Journal*, vol. 22, no. 7, 2009.

[9] G. Barton, *Smart Agriculture: Concepts and Applications*, Springer, 2020.

[10] J. Banks, J. S. Carson, B. L. Nelson, and D. M. Nicol, *Discrete-Event System Simulation*, 5th ed., Prentice Hall, 2010.

[11] "ESP32 Series Datasheet," Espressif Systems. [Online]. Available: https://www.espressif.com/en/products/socs/esp32

[12] Arduino, "Official Documentation and Projects," [Online]. Available: https://www.arduino.cc

[13] SparkFun Electronics, "Product Tutorials," [Online]. Available: https://www.sparkfun.com

[14] Adafruit Industries, "Learning System," [Online]. Available: https://learn.adafruit.com/

[15] TutorialsPoint, "Sensors and Actuators," [Online]. Available: https://www.tutorialspoint.com

[16] "RDM6300 RFID Reader Module Datasheet," Components101. [Online]. Available: https://components101.com/modules/rdm6300-rfid-reader-module

[17] "LCD 16x2 Interfacing with NodeMCU," ElectronicWings. [Online]. Available: https://www.electronicwings.com/nodemcu/lcd16x2-interfacing-with-nodemcu

[18] R. Santos, "ESP32 with Firebase Realtime Database," Random Nerd Tutorials. [Online]. Available: https://randomnerdtutorials.com/esp32-firebase-realtime-database/

[19] R. Santos, "Data Logging with Firebase," Random Nerd Tutorials. [Online]. Available: https://randomnerdtutorials.com/esp32-data-logging-firebase-realtime-database/

[20] R. Santos, "Firebase Web App Guide," Random Nerd Tutorials. [Online]. Available: https://randomnerdtutorials.com/esp32-firebase-web-app/

[21] Google Firebase, “Realtime Database,” [Online]. Available: https://firebase.google.com/docs/database

[22] Google Firebase, “Firebase Authentication,” [Online]. Available: https://firebase.google.com/docs/auth

[23] Mozilla Developer Network (MDN), “Web Development Documentation,” [Online]. Available: https://developer.mozilla.org/

[24] W3Schools, “HTML/CSS/JS Tutorials,” [Online]. Available: https://www.w3schools.com

[25] IEEE Xplore, “Smart Agriculture Papers,” [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp

[26] Elsevier, “Computers and Electronics in Agriculture,” [Online]. Available: https://www.sciencedirect.com

ΠΑΡΑΡΤΗΜΑ Α : ΚΩΔΙΚΑΣ ESP32

```
#include <LCD-I2C.h>
#include <Wire.h>
#include <RTClib.h>
#include <EEPROM.h>
#include <WiFi.h>
#include <WiFiManager.h>
#include <esp_wifi.h>
#include <esp_wps.h>
#include <Firebase_ESP_Client.h>
#include <SoftwareSerial.h>

SoftwareSerial readerIn(0, -1);  // RX, TX pins για reader εισόδου
SoftwareSerial readerOut(2, -1);   // RX, TX pins για reader εξόδου

String allowedTags[5] = {
  "31004187AB",
  "0D002567A7",
  "1800793145",
  "05003860F7",
  "3800C5A41D"
 };
// Κατάσταση για κάθε tag (true = μέσα, false = έξω)
bool insideStatus[5] = {false, false, false, false, false};

int MaxCknNo=4;

int chknNO=4;
int in=0;
int out=0;

#define I2C_SDA 21
#define I2C_SCL 22
#define EEPROM_SIZE 512
#define SSID_ADDR 8
#define PASS_ADDR 9
#define MAX_SSID_LEN 32
#define MAX_PASS_LEN 64
#define WIFI_TIMEOUT_MS 10000
#define WPS_MODE WPS_TYPE_PBC
esp_wps_config_t wpsConfig = WPS_CONFIG_INIT_DEFAULT(WPS_MODE);

#define           DATABASE_URL           "https://jmp-coop-default-rtdb.europe-
west1.firebasedatabase.app/"
#define API_KEY "AIzaSyDUMuRZCncdTnVhhVRHwDd_kjY2vJW6ST0"
#define USER_EMAIL "esp32@jmp.com"            // Εσύ το ορίζεις
#define USER_PASSWORD "12345678"

#define R_FIXED 10000.0
#define BETA 3950
#define T0 298.15
#define VCC 3.29

FirebaseData fbdo;
```

```
FirebaseAuth auth;
FirebaseConfig config;

unsigned long lastCommandTime = 0;
unsigned long lastKeepAlive = 0;
const unsigned long keepAliveInterval = 60000;                    // κάθε 60
δευτερόλεπτα

const int TempS = 1;
const int lightS = 3;
const int upb = 4;                                                //GPIO INPUT
BUTTON UP
const int down = 5;                                               //GPIO INPUT
BUTTON DOWN
const int selectb = 6;                                            //GPIO INPUT
BUTTON SELECT
const int waterlv = 7;                                            //GPIO WATER
LEVEL
const int OB2 = 8;                                               //GPIO MOTOR2
(DOOR)
const int OA2 = 10;                                              //GPIO MOTOR2
(DOOR)
const int HEATER = 11;
const int in2 = 12;                                              //GPIO MOTOR1
(FEEDER)
const int in1 = 13;                                              //GPIO MOTOR1
(FEEDER)
const int Dup = 15;
const int Ddown = 18;
const int wrnlight = 19;                                        //GPIO WARNING
LIGHT OUTPUT
const int foodlv = 20;                                            //GPIO FOOD
LEVEL
const int RainS = 23;

int tmrd=0;                                                      // XRONOS OPEN-
CLOSE AUTO PORTAS
int tmrdc=0;
int triedConnection = 0;                                         //FLAG GIA WIFI
CONNECTION
int wrn=0;
int ind=0;
int fd=0;                                                       //FLAG OTI TAISE
int dr=0;
int rpt=10;
int wifion=0;
int tmr1=0;
int tmrwth=0;
int menu=0;
int press=0;
int pressUP=0;
int pressDWN=0;
int doorA=0;                                                                //DOOR
AUTO ON-OFF
int doorST=0;                                                               //DOOR
POSITION 0PEN="1", CLOESED="0"
```

```
int feederST=0;                                                    //FEEDER
ON-OFF
int dur=0;                                                        //DURATION
MENU
int WthrPR=0;
int mnld=0;
int insideCount;
int outsideCount;
int lowf=0;
int loww=0;
int chkleftout=0;
int doorlck=0;
int bdwthr=0;

int setdM=0;
int posdM=0;
int setfM=0;
int posfM=0;
int posW=0;
int setWP=0;
int setRT=0;

int pos=0;
int posd=0;
int posf=0;
int posWP=0;

int fl=0;
int flt=0;
int flf=0;
int fla=0;
int flaf=0;
int flaW=0;
int flaWP=0;
int flaRT=0;

int AsetO=0;
int AsetC=0;
int set=0;
int setd=0;
int setf=0;
int setWf=0;

int sett=0;
int settd=0;
int settdc=0;
int settf=0;

int setHour = 0, setMin = 0;
int setHourOD = 0, setMinOD = 0;
int setHourCD = 0, setMinCD = 0;
int setHourAOD = 0, setMinAOD = 0;
int setHourACD = 0, setMinACD = 0;
int setHourF = 0, setMinF = 0;

RTC_DS1307 rtc;
```

```
LCD_I2C lcd(0x27, 16, 2); // Default address of most PCF8574 modules, change
according

void setup() {

    analogReadResolution(12);

    pinMode(upb, INPUT_PULLUP);           //using internal pullup
    pinMode(down, INPUT_PULLUP);           //using internal pullup
    pinMode(selectb, INPUT_PULLUP);
    pinMode(foodlv, INPUT_PULLUP);
    pinMode(waterlv, INPUT_PULLUP);
    pinMode(Dup, INPUT_PULLUP);
    pinMode(Ddown, INPUT_PULLUP);
    pinMode(lightS, INPUT);
    pinMode(TempS, INPUT);
    pinMode(RainS, INPUT);
    pinMode(in1, OUTPUT);
    pinMode(in2, OUTPUT);
    pinMode(OA2, OUTPUT);
    pinMode(OB2, OUTPUT);
    pinMode(wrnlight, OUTPUT);
    pinMode(HEATER, OUTPUT);

    Wire.begin(I2C_SDA, I2C_SCL);
    lcd.begin(&Wire);
    lcd.display();
    lcd.backlight();

    rtc.begin();
    Serial.begin(115200);

    readerIn.begin(9600);
    readerOut.begin(9600);

    WiFi.mode(WIFI_STA);
    digitalWrite(wrnlight,LOW);
    digitalWrite(HEATER,LOW);


    //           KATAXWRHSH-LEITOURGEIA EEPROM

    EEPROM.begin(EEPROM_SIZE);

if                  (EEPROM.read(0)==0xFF){doorA=0;}else{doorA=EEPROM.read(0);}
//AUTO OR MANUAL DOOR
if    (EEPROM.read(1)==0xFF){setHourOD    =    0;}else{setHourOD=EEPROM.read(1);}
//DOOR OPEN HOUR
if    (EEPROM.read(2)==0xFF){setMinOD    =    0;}else{setMinOD=EEPROM.read(2);}
//DOOR OPEN MINUTES
if    (EEPROM.read(3)==0xFF){setHourCD    =    0;}else{setHourCD=EEPROM.read(3);}
//DOOR CLOSE HOUR
if    (EEPROM.read(4)==0xFF){setMinCD    =    0;}else{setMinCD=EEPROM.read(4);}
//DOOR CLOSE MINUTES
if            (EEPROM.read(5)==0xFF){feederST=0;}else{feederST=EEPROM.read(5);}
//FEEDER ON OR OFF
```

```
if    (EEPROM.read(6)==0xFF){setHourF    =    0;}else{setHourF=EEPROM.read(6);}
//FEED HOUR
if     (EEPROM.read(7)==0xFF){setMinF     =     0;}else{setMinF=EEPROM.read(7);}
//FEED MINUTES
if (EEPROM.read(8)==0xFF){WthrPR = 0;}else{WthrPR=EEPROM.read(8);}
if (EEPROM.read(9)==0xFF){wifion = 0;}else{wifion=EEPROM.read(9);}
if (EEPROM.read(10)==0xFF){dur = 0;}else{dur=EEPROM.read(10);}
if (EEPROM.read(11)==0xFF){chknNO = 4;}else{chknNO=EEPROM.read(11);}
if (EEPROM.read(12)==0xFF){doorlck = 1;}else{doorlck=EEPROM.read(12);}

}


////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////
 ////////////////////////////////////////               ARXH PROGRAMMATOS LOOP
//////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////

void loop(){

    longPress();

    if  (  wifion==1  &&  triedConnection  ==  0){connectWithStoredCredentials();
triedConnection = 1;}
    if (wrn>0 && ind==0){digitalWrite(wrnlight,HIGH);ind=1;}else if(wrn==0 &&
ind==1){ digitalWrite(wrnlight,LOW); ind=0;}

    receiveIOT();
    Timers();
    tags();
    weatherProtect();
    warnings();

    lcdWRN();
    if   (tmr1>150   &&   tmr1<300   ){lcd.setCursor(0,   0);   lcd.print("JMp
Innovations");   lcd.setCursor(0, 1);   lcd.print("** SMART COOP **");}
    if (tmr1>300 && tmr1<450){lcdTimeTemp(); }
    if (tmr1>450 && tmr1<=600){lcdMSG(); }
    if (tmr1>600){SerialMSG(); lcd.clear(); tmr1=0; }

    delay(10);
    tmr1++;
}

////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////
 ////////////////////////////////////////               TELOS PROGRAMMATOS MAIN
//////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////
```

```
 void lcdTimeTemp(){
  float temperatureC = temp(TempS);
        DateTime now = rtc.now();
        lcd.setCursor(0, 1);
        lcd.printf(" %02d:%02d |", now.hour(), now.minute());
        lcd.print(" TEMP:");
        lcd.print(temperatureC);
 }



void lcdWRN(){

    if (tmr1<50 && lowf==1){lcd.setCursor(0, 0); lcd.print("!!  WARNING  !!");
lcd.setCursor(0, 1); lcd.print("LOW  FOOD  LEVEL"); }
    if (tmr1>=50 && tmr1<100 && loww==1){lcd.setCursor(0, 0); lcd.print("!!
WARNING  !!"); lcd.setCursor(0, 1); lcd.print("LOW WATER  LEVEL"); }
    if   (tmr1>=100   &&   tmr1<150   &&   chkleftout==1){lcd.setCursor(0,   0);
lcd.print("!!  WARNING  !!"); lcd.setCursor(0, 1); lcd.print("  CHICKEN OUT  ");
}

}

void warnings(){

  if (digitalRead(foodlv) == HIGH ){wrn=1; lowf=1;} else  lowf=0;
  if (digitalRead(waterlv) == HIGH ){wrn=1; loww=1;} else  loww=0;
  if (doorST==0 && in==0){wrn=1; chkleftout=1; } else  chkleftout=0;
  if (lowf==0 && loww==0 && chkleftout==0){wrn=0;}


}

 void lcdMSG(){

        lcd.setCursor(0, 0);
        lcd.print("     KOTES     ");
        lcd.setCursor(0, 1);
        if (insideCount == chknNO) {lcd.print(" OLES EINAI MESA  ");} else if
(outsideCount == chknNO) {lcd.print(" OLES EINAI EXW  ");}else{
        lcd.print(" MESA: ");
        lcd.print(insideCount);
        lcd.print("  EXW: ");
        lcd.print(outsideCount);}


 }

 void SerialMSG(){

      float temperatureC = temp(TempS);

      if(digitalRead(Dup)==LOW){doorST=1;}
      if(digitalRead(Ddown)==LOW){doorST=0;}

        DateTime now = rtc.now();
```

```
        Serial.print(" Time: " );
        Serial.print(now.hour());
        Serial.print(":");
        Serial.print(now.minute());
        if (doorA==1) {Serial.print(" | Door is AUTO ");}else if (doorlck==1)
{Serial.print(" | Door is LOCKED ");}else{
        Serial.print(" | Door open at: " );
        Serial.print(setHourOD);
        Serial.print(":");
        Serial.print(setMinOD);
        Serial.print(" | Door close at: " );
        Serial.print(setHourCD);
        Serial.print(":");
        Serial.print(setMinCD);}
        if (feederST==0){Serial.print(" | Feeder is OFF " );}else{
        Serial.print(" | Feed at: " );
        Serial.print(setHourF);
        Serial.print(":");
        Serial.print(setMinF);}
        if    (WthrPR==0){Serial.print("    |    Weather    Protection    is
OFF");}else{Serial.print(" | Weather Protection is ON");}
        Serial.print(" | Temperature(ºC): ");
        Serial.print(temperatureC);
         Serial.print(" | Door status : ");
        if   (digitalRead(Ddown)==0)   {   Serial.print("   CLOSED   ");}else
if(digitalRead(Dup)==LOW){Serial.print(" OPEN ");}
       if (insideCount == chknNO) {Serial.print(" | ΟΛΕΣ ΟΙ ΚΟΤΕΣ ΕΙΝΑΙ ΜΕΣΑ");}
else  if  (outsideCount  ==  chknNO)  {Serial.print("  |  ΟΛΕΣ  ΟΙ  ΚΟΤΕΣ  ΕΙΝΑΙ
ΕΞΩ");}else{
        Serial.print(" | ΚΟΤΕΣ ΜΕΣΑ: ");
        Serial.print(insideCount);
        Serial.print(" | ΚΟΤΕΣ ΕΞΩ : ");
        Serial.print(outsideCount);}
        Serial.print(" | status : ");
        if (wrn==0) { Serial.print(" NORNAL ");}else{Serial.print(" WARNING ");}
        if (WiFi.status() == WL_CONNECTED) { Serial.print(" | Connected to: " +
WiFi.SSID());}else{Serial.print(" | Disconnected ");}

        Serial.println();

         // Αντικείμενο JSON για αποστολή
  FirebaseJson json;
  json.set("hour", now.hour());
  json.set("minute", now.minute());
  json.set("AUTO DOOR", doorA);
  json.set("setHourOD", setHourOD);
  json.set("setMinOD", setMinOD);
  json.set("setHourCD", setHourCD);
  json.set("setMinCD", setMinCD);
  json.set("FEEDER", feederST);
  json.set("setHourF", setHourF);
  json.set("setMinF", setMinF);
  json.set("WEATHER PROTECTION", WthrPR);
  json.set("WARNINGS", wrn);
  json.set("Chiken Inside", insideCount);
  json.set("Chiken Outside", outsideCount);
```

```
  json.set("Chiken all inside", in);
  json.set("Chiken all Outside", out);
  json.set("Door Position", doorST);
  json.set("temperature", temperatureC);
  json.set("DOOR STATUS", doorlck);

  // Στείλε τα δεδομένα στο path "/esp32/data"
  if (Firebase.RTDB.setJSON(&fbdo, "/esp32/data", &json)) {
    Serial.println("Data sent to Firebase");
  } else {keepConnected();
    Serial.print("Firebase error: ");
    Serial.println(fbdo.errorReason());
  }
 }

void receiveIOT(){
  if (millis() - lastCommandTime > 2000) {
    lastCommandTime = millis();

    if (Firebase.RTDB.getJSON(&fbdo, "/esp32/commands")) {
      FirebaseJson &json = fbdo.jsonObject();
      FirebaseJsonData cmdType;
      if (json.get(cmdType, "type")) {
        if (cmdType.stringValue == "manual_feed") {
          Serial.println("Received manual_feed command!");
          FEED();
          clearCommand(); }
        else if (cmdType.stringValue == "toggle_door") {
          Serial.println("Received manual open/close Door command!");
          doorOpenClose();
          clearCommand(); }
        else if (cmdType.stringValue == "update_settings") {
          FirebaseJsonData weatherVal, feederVal, autoDoorVal;
          FirebaseJsonData temp;
                    // Weather protection
          if (json.get(weatherVal, "weather")) {
            int remoteWthrPR = weatherVal.boolValue ? 1 : 0;
            if (WthrPR != remoteWthrPR) {
              WthrPR = remoteWthrPR;
              Serial.println("Weather protection updated remotely: "); }}
                    // Feeder
          if (json.get(feederVal, "feeder")) {
            int remoteFeeder = feederVal.boolValue ? 1 : 0;
            if (feederST != remoteFeeder) {
              feederST = remoteFeeder;
              Serial.println("feeder on/off updated remotely: "); }}
                   // Auto Door
          if (json.get(autoDoorVal, "autoDoor")) {
            int remoteDoor = autoDoorVal.boolValue ? 1 : 0;
            if (doorA != remoteDoor) {
              doorA = remoteDoor;
              Serial.print("Auto Door updated updated remotely: ");}}
                              // Feed Time
          if (json.get(temp, "feederHour")) setHourF = temp.intValue;
          if (json.get(temp, "feederMin"))  setMinF  = temp.intValue;
                             // Open Door Time
```

```
          if (json.get(temp, "openHour"))   setHourOD = temp.intValue;
          if (json.get(temp, "openMin"))    setMinOD  = temp.intValue;
                              // Close Door Time
          if (json.get(temp, "closeHour"))  setHourCD = temp.intValue;
          if (json.get(temp, "closeMin"))   setMinCD  = temp.intValue;
          clearCommand();}
      }
    } else {Serial.println("Failed to get command: " + fbdo.errorReason());}
  }
}


void clearCommand() {
  FirebaseJson empty;
  empty.set("type", "none");
  Firebase.RTDB.setJSON(&fbdo, "/esp32/commands", &empty);
}

void Timers(){

  DateTime now = rtc.now();
    if  ((now.hour()==setHourOD)  &&  (now.minute()==setMinOD)  &&  doorA==0  &&
dr==0){doorOpenClose(); dr=1;}              //TIMING OPEN
    if     ((now.hour()==setHourOD)     &&     (now.minute()==setMinOD+1)     &&
doorA==0){dr=0;}
    if  ((now.hour()==setHourCD)  &&  (now.minute()==setMinCD)  &&  doorA==0  &&
dr==0){doorOpenClose(); dr=1;}              //TIMING CLOSE
    if     ((now.hour()==setHourCD)     &&     (now.minute()==setMinCD+1)     &&
doorA==0){dr=0;}
    if  ((now.hour()==setHourF)  &&  (now.minute()==setMinF)  &&  feederST==1  &&
fd==0){FEED(); fd=1;}       // FEED TIME
    if    ((now.hour()==setHourF)    &&    (now.minute()==(setMinF+1))){fd=0;}
//MHDENIZEI TO FEED FLAG
    if (doorA==1){AutoDoor();}
}

void weatherProtect(){

  if (WthrPR==1){
    float temperatureC = temp(TempS);
    if((digitalRead(RainS) == HIGH ||  temperatureC<=-1) && doorST==0){tmrwth=0;
doorlck=1; bdwthr=1;}else{tmrwth++;}
    if   (tmrwth>=100){doorlck=0;   if(analogReadMilliVolts(lightS)<=1500   &&
digitalRead(Ddown)==LOW && bdwthr==1){doorOpenClose(); bdwthr=0;} }
  if
(temperatureC<=0){digitalWrite(HEATER,HIGH);}else{digitalWrite(HEATER,LOW);}
  }
}

void doorOpenClose(){

  if (doorlck==0){
  mnld=1;
  while (doorST==0 && mnld==1){
        digitalWrite(OA2,HIGH);
        digitalWrite(OB2,LOW);
```

```
  if (digitalRead(Dup)==LOW){
      digitalWrite(OA2,LOW);
      digitalWrite(OB2,LOW);
      doorST=1;
      mnld=0;}}
   while (doorST==1 && mnld==1){
        digitalWrite(OA2,LOW);
        digitalWrite(OB2,HIGH);
   if(digitalRead(Ddown)==LOW){
      digitalWrite(OA2,LOW);
      digitalWrite(OB2,LOW);
      doorST=0;
      mnld=0;}}}
  else{lcd.clear(); lcd.setCursor(0, 0); lcd.print("DOOR LOCKED"); delay(1000);
lcd.clear(); }
      }

void FEED(){

for (int i=0; i<=rpt; i++){


  digitalWrite(in1,HIGH);
  digitalWrite(in2,LOW);

  delay(400);

  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);

  delay(200);
  }
  digitalWrite(in1,LOW);
  digitalWrite(in2,LOW);

}

void AutoDoor(){
  if    (analogReadMilliVolts(lightS)<=2000    &&    (digitalRead(Ddown)==LOW)){
tmrd++;}else{tmrd=0; AsetO=0;}
  if                    (analogReadMilliVolts(lightS)>2000                    &&
(digitalRead(Dup)==LOW)){tmrdc++;}else{tmrdc=0; AsetC=0;}
  if   (analogReadMilliVolts(lightS)<=2000   &&   (digitalRead(Ddown)==LOW)   &&
(tmrd>=300)){doorOpenClose(); }                //TIMING OPEN AFTER 15MIN
  if    (analogReadMilliVolts(lightS)>2000    &&    (digitalRead(Dup)==LOW)    &&
(tmrdc>=300)){doorOpenClose(); }                   //TIMING CLOSE AFTER 15MIN
}

void doorMode(){
  lcd.clear();
  lcd.print("   DOOR AUTO:  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){fla=1;}else{fla=0;}
  delay(100);
  while (setdM==1 && fla==1){
```

```
            if                                          (digitalRead(down)==LOW){if
(posdM==1){posdM=0;}else{posdM++;}delay(200);}
            if       (digitalRead(upb)==LOW){if       (posdM==0){posdM=1;}else{
posdM=posdM-1;}delay(200);}
            delay(50);
         if (posdM==0){
            lcd.setCursor(0, 1);
            lcd.print("    AUTO ON     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ doorA=1; fla=0; setdM=0;}
            }
         if ( posdM==1){
            lcd.setCursor(0, 1);
            lcd.print("    AUTO OFF     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ doorA=0; fla=0; setdM=0;}
            }
            }
            lcd.setCursor(0, 0);
            lcd.print("   DOOR MENU:  ");
            settd=0;
            posd=2;
}

void setDoorClosTime(){
   lcd.clear();
  lcd.print("SET CLOSE TIME:");
  delay(100);
  while (settdc=1 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("HOUR: %02d", setHourCD);
    if (digitalRead(upb) == LOW) {
      setHourCD = (setHourCD + 1) % 24;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setHourCD = (setHourCD - 1 + 24) % 24;
      delay(200);
    }
  }
   delay(1000);
   settdc=2;
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("SET MINUTES:");
  delay(100);
  while (settdc=2 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("MIN: %02d", setMinCD);
    if (digitalRead(upb) == LOW) {
      setMinCD = (setMinCD + 1) % 60;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setMinCD = (setMinCD - 1 + 60) % 60;
      delay(200);
```

```
    }
    }
    settdc=0;
    posd=1;
    lcd.setCursor(0, 0);
    lcd.print("   DOOR MENU:  ");
}

void setDoorOpenTime(){
  lcd.clear();
  lcd.print("SET OPEN TIME:");
  delay(100);
  while (settd=1 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("HOUR: %02d", setHourOD);
    if (digitalRead(upb) == LOW) {
      setHourOD = (setHourOD + 1) % 24;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setHourOD = (setHourOD - 1 + 24) % 24;
      delay(200);
    }
  }
   delay(1000);
   settd=2;
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("SET MINUTES:");
  delay(100);
  while (settd=2 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("MIN: %02d", setMinOD);
    if (digitalRead(upb) == LOW) {
      setMinOD = (setMinOD + 1) % 60;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setMinOD = (setMinOD - 1 + 60) % 60;
      delay(200);
    }
    }
    settd=0;
    posd=0;
    lcd.setCursor(0, 0);
    lcd.print("   DOOR MENU:  ");
}

void doorMenu(){
  lcd.clear();
  lcd.print("   DOOR MENU:  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){fl=2;}else{fl=0;}
  delay(100);
  while (setd==1 && fl==2){
```

```
            if                                          (digitalRead(down)==LOW){if
(posd==4){posd=0;}else{posd++;}delay(200);}
            if  (digitalRead(upb)==LOW){if  (posd==0){posd=4;}else{  posd=posd-
1;}delay(200);}

         delay(50);
         if (posd==0){
            lcd.setCursor(0, 1);
            lcd.print(" OPEN TIME SET ");
            delay(100);

            if (digitalRead(selectb)==LOW){ settd=1;}else{settd=0;}
            if (settd==1){doorlck=0; setDoorOpenTime();}

            }
         if ( posd==1){
            lcd.setCursor(0, 1);
            lcd.print(" CLOSE TIME SET ");
            delay(100);

            if (digitalRead(selectb)==LOW){ settdc=1;}else{settdc=0;}
            if (settdc==1){doorlck=0; setDoorClosTime();}

            }
        if ( posd==2){
            lcd.setCursor(0, 1);
            lcd.print("      AUTO      ");
            delay(100);

            if (digitalRead(selectb)==LOW){ setdM=1; doorlck=0; doorMode();}

            }

         if ( posd==3){
            lcd.setCursor(0, 1);
            lcd.print("  DOOR LOCK  ");
            if   (digitalRead(selectb)==LOW){   pos=1;posd=0;   setd=0;   fl=0;
doorlck=1; delay(500);
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU ");}

            }
         if ( posd==4){
            lcd.setCursor(0, 1);
            lcd.print("      BACK      ");
            if   (digitalRead(selectb)==LOW){   pos=1;posd=0;   setd=0;   fl=0;
delay(500);
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU ");}

            if (doorA!=EEPROM.read(0)){
            EEPROM.write(0,doorA);
            EEPROM.commit();}
            if (setHourOD!=EEPROM.read(1)){
            EEPROM.write(1,setHourOD);
            EEPROM.commit();}
```

```
            if (setMinOD!=EEPROM.read(2)){
            EEPROM.write(2,setMinOD);
            EEPROM.commit();}
            if (setHourCD!=EEPROM.read(3)){
            EEPROM.write(3,setHourCD);
            EEPROM.commit();}
            if (setMinCD!=EEPROM.read(4)){
            EEPROM.write(4,setMinCD);
            EEPROM.commit();}
            if (doorlck!=EEPROM.read(12)){
            EEPROM.write(12,doorlck);
            EEPROM.commit();}
            }
  }
 }

void setFeedTime(){

  lcd.clear();
  lcd.print("SET FEED TIME:");
  delay(100);
  while (settf=1 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("HOUR: %02d", setHourF);
    if (digitalRead(upb) == LOW) {
      setHourF = (setHourF + 1) % 24;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setHourF = (setHourF - 1 + 24) % 24;
      delay(200);
    }
  }
   delay(1000);
   settf=2;
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("SET MINUTES:");
  delay(100);
  while (settf=2 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("MIN: %02d", setMinF);
    if (digitalRead(upb) == LOW) {
      setMinF = (setMinF + 1) % 60;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setMinF = (setMinF - 1 + 60) % 60;
      delay(200);
    }
    }
    delay(1000);
   settf=3;
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print(" SET DURATION :");
```

```
  delay(100);
  while (settf=3 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("PROGRAM: %02d", dur);
    if (digitalRead(upb) == LOW) {
      dur = (dur + 1) % 3;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      dur = (dur - 1 + 3) % 3;
      delay(200);
    }
    }
    if (dur==0){rpt=10;}
    else if (dur==1){rpt=20;}
    else if (dur==2){rpt=30;}
    settf=0;
    posf=0;
    lcd.setCursor(0, 0);
    lcd.print("   FEEDER MENU:  ");
}

void feedMode(){
  lcd.clear();
  lcd.print("   FEEDER ON/OFF:  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){flaf=1;}else{flaf=0;}
  delay(100);
  while (setfM==1 && flaf==1){
            if                                        (digitalRead(down)==LOW){if
(posfM==1){posfM=0;}else{posfM++;}delay(200);}
            if       (digitalRead(upb)==LOW){if       (posfM==0){posfM=1;}else{
posfM=posfM-1;}delay(200);}
            delay(50);
         if (posfM==0){
            lcd.setCursor(0, 1);
            lcd.print("    FEEDER ON     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ feederST=1; flaf=0; setfM=0;}
            }
         if ( posfM==1){
            lcd.setCursor(0, 1);
            lcd.print("    FEEDER OFF     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ feederST=0; flaf=0; setfM=0;}
            }
            }
            lcd.setCursor(0, 0);
            lcd.print("  FEEDER MENU:  ");
            posf=1;

}

void feederMenu(){
  lcd.clear();
  lcd.print("  FEEDER MENU:  ");
```

```
  delay(100);
  if (digitalRead(selectb) == HIGH ){flf=1;}else{flf=0;}
  delay(100);
  while (setf==1 && flf==1){
            if                                          (digitalRead(down)==LOW){if
(posf==3){posf=0;}else{posf++;}delay(200);}
            if  (digitalRead(upb)==LOW){if  (posf==0){posf=3;}else{  posf=posf-
1;}delay(200);}

         delay(50);
         if (posf==0){
            lcd.setCursor(0, 1);
            lcd.print("  SET FEED TIME  ");
            delay(100);

            if (digitalRead(selectb)==LOW){ settf=1;}else{settf=0;}
            if (settf==1){setFeedTime();}

            }
         if ( posf==1){
            lcd.setCursor(0, 1);
            lcd.print("   ON/OFF SET   ");
            delay(100);

            if (digitalRead(selectb)==LOW){
            setfM=1;
            feedMode();}

            }

         if ( posf==2){
            lcd.setCursor(0, 1);
            lcd.print("      BACK      ");
            if   (digitalRead(selectb)==LOW){   pos=2;posf=0;   setf=0;   flf=0;
delay(500);
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU ");}

            if (feederST!=EEPROM.read(5)){
            EEPROM.write(5,feederST);
            EEPROM.commit();}
            if (setHourF!=EEPROM.read(6)){
            EEPROM.write(6,setHourF);
            EEPROM.commit();}
            if (setMinF!=EEPROM.read(7)){
            EEPROM.write(7,setMinF);
            EEPROM.commit();}
            if (dur!=EEPROM.read(10)){
            EEPROM.write(10,dur);
            EEPROM.commit();}
            }
   }
 }

 void WeatherMenu(){
  lcd.clear();
```

```
  lcd.print(" WEATHER PROTECT  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){flaWP=1;}else{flaWP=0;}
  delay(100);
  while (setWP==1 && flaWP==1){
            if                                        (digitalRead(down)==LOW){if
(posWP==1){posWP=0;}else{posWP++;}delay(200);}
            if       (digitalRead(upb)==LOW){if       (posWP==0){posWP=1;}else{
posWP=posWP-1;}delay(200);}
            delay(50);
         if (posWP==0){
            lcd.setCursor(0, 1);
            lcd.print(" PROTECTION  ON     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ WthrPR=1; flaWP=0; setWP=0;}
            }
         if ( posWP==1){
            lcd.setCursor(0, 1);
            lcd.print(" PROTECTION OFF     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ WthrPR=0; flaWP=0; setWP=0;}
            }
            }
            if
(WthrPR!=EEPROM.read(8)){EEPROM.write(8,WthrPR);EEPROM.commit();}
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU  ");
            posWP=0;

 }


//////////////////////
// Connect with saved WiFi
//////////////////////
void connectWithStoredCredentials() {

  WiFiManager wifiManager;
  wifiManager.autoConnect();

    auth.user.email = USER_EMAIL;
    auth.user.password = USER_PASSWORD;
    config.api_key = API_KEY;
    config.database_url = DATABASE_URL;

    Firebase.begin(&config, &auth);
    Firebase.reconnectWiFi(true);

}

void keepConnected(){
  if (!Firebase.ready()) {
    auth.user.email = USER_EMAIL;
    auth.user.password = USER_PASSWORD;
    config.api_key = API_KEY;
    config.database_url = DATABASE_URL;
```

```
    Firebase.begin(&config, &auth);
    Firebase.reconnectWiFi(true);
  }
  }

//////////////////////
// Connect with WPS PROSORINA OFF LOGO ASUMVATOU MODULE
//////////////////////
void connectWithWPS() {
  Serial.println("📡 Starting WPS...");

  WiFi.disconnect(true);
  delay(1000);
  WiFi.mode(WIFI_MODE_STA);

  esp_wifi_wps_disable();
  esp_wifi_wps_enable(&wpsConfig);
  esp_wifi_wps_start(0);

  WiFi.onEvent([](WiFiEvent_t event, arduino_event_info_t info) {
    if (event == ARDUINO_EVENT_WPS_ER_SUCCESS) {
      String ssid = WiFi.SSID();
      String pass = WiFi.psk();
      Serial.println("☑ WPS Success!");
      Serial.print("📶 SSID: "); Serial.println(ssid);
      Serial.print("🔑 PASS: "); Serial.println(pass);

  //    writeEEPROMString(SSID_ADDR, ssid);
  //    writeEEPROMString(PASS_ADDR, pass);
      WiFi.begin(ssid.c_str(), pass.c_str());
    }  else  if  (event  ==  ARDUINO_EVENT_WPS_ER_FAILED  ||  event  ==
ARDUINO_EVENT_WPS_ER_TIMEOUT) {
      Serial.println("❌ WPS Failed or Timed out.");
      esp_wifi_wps_disable();
    }
  });
}

//////////////////////
// Connect with WiFiManager
//////////////////////
void connectWithWiFiManager() {
  WiFiManager wm;
  Serial.println("🌐 Starting WiFiManager portal...");

  if (wm.autoConnect("ESP32_Config")) {
    Serial.println("☑ Connected via WiFiManager!");
   WiFi.begin(WiFi.SSID().c_str(), WiFi.psk().c_str());
  Serial.println(WiFi.localIP());
  } else {
    Serial.println("❌ WiFiManager failed. Restarting...");
    ESP.restart();
  }
}
```

```
//////////////////////
// Disconnect and Clear EEPROM
//////////////////////
void disconnectAndForgetWiFi() {
  Serial.println("🚫 Disconnecting and clearing saved credentials...");

  WiFi.disconnect(true, true);

  Serial.println("☑ WiFi credentials cleared.");
  triedConnection = 0;
}


void WiFimenu(){
  lcd.clear();
  lcd.print(" WiFi SETTINGS:  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){flaW=1;}else{flaW=0;}
  delay(100);
  while (setWf==1 && flaW==1){
            if                                        (digitalRead(down)==LOW){if
(posW==2){posW=0;}else{posW++;}delay(200);}
            if  (digitalRead(upb)==LOW){if  (posW==0){posW=2;}else{  posW=posW-
1;}delay(200);}
            delay(50);
         if (posW==8){
            lcd.setCursor(0, 1);
            lcd.print("   WIFI W WPS    ");
            delay(100);
            if    (digitalRead(selectb)==LOW){    connectWithWPS();    flaW=0;
setWf=0;wifion=1;}
            }
        if (posW==0){
            lcd.setCursor(0, 1);
            lcd.print("   WIFI W PHONE    ");
            delay(100);
            if  (digitalRead(selectb)==LOW){  connectWithWiFiManager();  flaW=0;
setWf=0;wifion=1;}
            }
         if ( posW==1){
            lcd.setCursor(0, 1);
            lcd.print("    DISC0NECT     ");
            delay(100);
            if (digitalRead(selectb)==LOW){ disconnectAndForgetWiFi(); flaW=0;
setWf=0;wifion=0;}
            }
         if ( posW==2){
            lcd.setCursor(0, 1);
            lcd.print("      BACK       ");
            delay(100);
            if (digitalRead(selectb)==LOW){flaW=0; setWf=0;}
            }
            }
            if
(wifion!=EEPROM.read(9)){EEPROM.write(9,wifion);EEPROM.commit();}
```

```
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU ");
            posW=0;
            pos=5;
}

void TagsSet(){
  lcd.clear();
  lcd.print("TOTAL CHICKENS: :  ");
  delay(100);
  if (digitalRead(selectb) == HIGH ){flaRT=1;}else{flaRT=0;}
  delay(100);
  while (setRT==1 && (digitalRead(selectb) == HIGH ) && flaRT==1){
    lcd.setCursor(0, 1);
    lcd.printf(" NUMBER : %02d", chknNO);
    if (digitalRead(upb) == LOW) {
      chknNO = (chknNO + 1) % (MaxCknNo+1);
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      chknNO = (chknNO - 1 + (MaxCknNo+1)) % (MaxCknNo+1);
      delay(200);
    }
    }                                                                    if
(chknNO!=EEPROM.read(11)){EEPROM.write(11,chknNO);EEPROM.commit();}
            lcd.setCursor(0, 0);
            lcd.print(" SETTINGS MENU ");
            pos=4;
            setRT=0;
}


void setHourMenu(){

  DateTime now = rtc.now();
  setHour = now.hour();

  lcd.clear();
  lcd.print("SET TIME:");
  delay(100);
  while (sett=1 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("HOUR: %02d", setHour);
    if (digitalRead(upb) == LOW) {
      setHour = (setHour + 1) % 24;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setHour = (setHour - 1 + 24) % 24;
      delay(200);
    }
  }
   rtc.adjust(DateTime(now.year(),     now.month(),     now.day(),     setHour,
now.minute(), 0));
   delay(300);
   sett=2;
```

```
   fl=0;
   pos=0;
   lcd.setCursor(0, 0);
  lcd.print(" SETTINGS MENU ");
}

void setMinuteMenu() {
  DateTime now = rtc.now();
  setMin = now.minute();
  lcd.clear();
  lcd.print("SET MINUTES:");
  delay(100);
  while (sett=1 && (digitalRead(selectb) == HIGH )){
    lcd.setCursor(0, 1);
    lcd.printf("MIN: %02d", setMin);
    if (digitalRead(upb) == LOW) {
      setMin = (setMin + 1) % 60;
      delay(200);
    }
    if (digitalRead(down) == LOW) {
      setMin = (setMin - 1 + 60) % 60;
      delay(200);
    }
    }
    rtc.adjust(DateTime(now.year(), now.month(), now.day(), now.hour(), setMin,
0));
    delay(300);
    sett=0;
    fl=0;
    pos=0;
    lcd.setCursor(0, 0);
    lcd.print(" SETTINGS MENU ");
    }

    void longPress(){

        if (digitalRead(selectb)==LOW){
          press++;
          if (digitalRead(selectb)==LOW && press>=100){
          set=1;
          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print(" SETTINGS MENU ");
          }

        while (set==1){ settings();}
    }

        if (digitalRead(upb)==LOW){
        pressUP++;
          if (digitalRead(upb)==LOW && pressUP>=100){

           lcd.clear();
           lcd.setCursor(0, 0);
           lcd.print("   MANUAL   ");
           lcd.setCursor(0, 1);
```

```
          lcd.print("   FEEDING   ");
          delay(500);
          FEED();
          pressUP=0;

        }
      }
        if (digitalRead(down)==LOW){
          pressDWN++;
          if (digitalRead(down)==LOW && pressDWN>=100){

          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print(" MANUAL DOOR ");
          lcd.setCursor(0, 1);
          lcd.print("   OPEN-CLOSE    ");
          delay(500);

         doorOpenClose();

          pressDWN=0;

        }
      }
    }

    void settings(){
            if                                        (digitalRead(down)==LOW){if
(pos==6){pos=0;}else{pos++;}delay(100);}
            if   (digitalRead(upb)==LOW){if   (pos==0){pos=6;}else{   pos=pos-
1;}delay(100);}
            if                                      (digitalRead(selectb)==HIGH){
fl=1;delay(100);}else{fl==0;delay(100);}
         delay(100);

         if (pos==0){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("     TIME     ");

              if                                    (digitalRead(selectb)==HIGH){
flt=1;delay(100);}else{flt==0;delay(100);}
              if (digitalRead(selectb)==LOW && flt==1){ sett=1;}else{sett=0;}
              if (sett==1){setHourMenu();}
              if (sett==2){setMinuteMenu();}

            }
         if (pos==1){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("     DOOR     ");
              if (digitalRead(selectb)==LOW){
              setd=1;
              doorMenu();
             }
```

```
            }
         if ( pos==2){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("     FEEDER     ");
              if (digitalRead(selectb)==LOW){
              setf=1;
              feederMenu();}
            }

            if ( pos==3){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print(" WEATHER PROTECT  ");
            if (digitalRead(selectb)==LOW){
              setWP=1;
              WeatherMenu();}

            }
         if ( pos==4){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("      RFID      ");
            if (digitalRead(selectb)==LOW){
             setRT=1;
            TagsSet();}
            }
         if ( pos==5){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("     WI-FI      ");
            if (digitalRead(selectb)==LOW){
            setWf=1;
            WiFimenu();}
            }
         if (pos==6){
            lcd.setCursor(0,  1);  //  Or  setting  the  cursor  in  the  desired
position.
            lcd.print("     BACK     ");

            if (digitalRead(selectb)==LOW){ pos=0; set=0; delay(500);}

            }
         press=0;
         }


float adcToVoltage(int adc) {
  return adc * (1.60 / 1600.0);  // Calibration με βάση τη μέτρηση
}

float temp(int pin) {
  int adcValue = analogRead(pin);
  float voltage = adcToVoltage(adcValue);

  float rNTC = R_FIXED * ((VCC - voltage) / voltage);
```

```
  float temperatureK = 1.0 / (1.0/T0 + (1.0/BETA) * log(rNTC / 10000.0));
  return temperatureK - 273.15;
}

void tags(){
  if (readerIn.available()) {
    String tag = readTag(readerIn);
    int idx = findTagIndex(tag);
    if (idx != -1) {
      insideStatus[idx] = true;  // Το tag μπήκε μέσα
      printStatus();
    }
  }

  if (readerOut.available()) {
    String tag = readTag(readerOut);
    int idx = findTagIndex(tag);
    if (idx != -1) {
      insideStatus[idx] = false;  // Το tag βγήκε έξω
      printStatus();
    }
  }
}

String readTag(SoftwareSerial &reader) {
  // Διαβάζει 12 χαρακτήρες από τον reader (RDM6300 στέλνει 12 bytes)
  String tagData = "";
  unsigned long start = millis();
  while (tagData.length() < 10 && (millis() - start) < 5) {
    if (reader.available()) {
      char c = reader.read();
      if (isHexadecimalDigit(c)) {
        tagData += c;
      }
    }
  }
  return tagData;
}

int findTagIndex(String tag) {
  for (int i = 0; i < chknNO; i++) {
    if (allowedTags[i].equalsIgnoreCase(tag)) {
      return i;
    }
  }
  return -1;
}

void printStatus() {
  insideCount = 0;
  for (int i = 0; i < chknNO; i++) {
    if (insideStatus[i]) insideCount++;
  }
  outsideCount = chknNO - insideCount;
```

```
Serial.print("ΚΟΤΕΣ ΜΕΣΑ: ");
Serial.print(insideCount);
Serial.print(",ΚΟΤΕΣ ΕΞΩ : ");
Serial.println(outsideCount);

if (insideCount == chknNO) {
  Serial.println("ΟΛΕΣ ΟΙ ΚΟΤΕΣ ΕΙΝΑΙ ΜΕΣΑ");
  in=1;
  out=0;
} else if (outsideCount == chknNO) {
  Serial.println("ΟΛΕΣ ΟΙ ΚΟΤΕΣ ΕΙΝΑΙ ΕΞΩ!");
  out=1;
  in=0;
}else {in=0; out=0;}}
```

ΠΑΡΑΡΤΗΜΑ Β : ΚΩΔΙΚΑΣ HTML

```
<!DOCTYPE html>
<html lang="el">
<head>
  <meta charset="UTF-8">
  <title>ESP32 Coop By JMp</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="apple-touch-icon" href="/coop.png">
  <style>
    body { font-family: sans-serif; padding: 20px; background: #f0f0f0; }
    h1, h2, h3 { color: #222; text-align: center; }
    .card {
      background: white; padding: 20px;
      border-radius: 10px; margin-bottom: 20px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
    }
    .flex-row {
      display: flex;
      flex-wrap: wrap;
      gap: 20px;
      align-items: center;
      justify-content: center;
      margin-bottom: 15px;
    }
    label { font-weight: bold; }

    input[type="number"] {
      width: 60px; padding: 5px;
      margin-right: 10px; margin-top: 5px;
    }

    .switch {
```

```
  position: relative; display: inline-block;
  width: 50px; height: 24px;
}
.switch input { opacity: 0; width: 0; height: 0; }
.slider {
  position: absolute; cursor: pointer;
  top: 0; left: 0; right: 0; bottom: 0;
  background-color: #ccc; transition: .4s;
  border-radius: 24px;
}
.slider:before {
  position: absolute; content: "";
  height: 18px; width: 18px; left: 3px; bottom: 3px;
  background-color: white; transition: .4s;
  border-radius: 50%;
}
input:checked + .slider { background-color: #4caf50; }
input:checked + .slider:before { transform: translateX(26px); }

button {
  margin-top: 10px; padding: 10px 20px;
  background: #007bff; color: white;
  border: none; border-radius: 5px; cursor: pointer;
}
button:hover { background: #0056b3; }

.danger-button {
  background-color: red;
  color: white;
  border-radius: 50%;
  width: 60px;
  height: 60px;
```

```
  font-size: 12px;
  font-weight: bold;
  margin-right: 10px;
}

.warning {
  font-weight: bold;
  padding: 10px;
  border-radius: 5px;
  color: white;
  background-color: red;
  animation: blink 1s infinite;
  text-align: center;
}

@keyframes blink {
  0%, 50%, 100% { opacity: 1; }
  25%, 75% { opacity: 0; }
}

#page-content {
  display: none;
}

#password-protection {
    text-align: center;
    margin-top: 50px;
}

#countdown-timer {
    position: fixed;
    bottom: 10px;
```

```
        right: 10px;
        background-color: rgba(0, 0, 0, 0.7);
        color: white;
        padding: 5px 10px;
        border-radius: 5px;
        display: none; /* Initially hidden */
    }
    .chicken-status-box {
        border: 1px solid #ccc;
        padding: 10px;
        margin-top: 10px;
        border-radius: 5px;
        background-color: #e9e9e9; /* Light grey background */
        text-align: center;
    }
  </style>
</head>
<body>

<div id="password-protection">
    <h2>Enter Password</h2>
    <input type="password" id="password-input">
    <button onclick="checkPassword()">Enter</button>
    <p id="error-message" style="color: red;"></p>
</div>

<div id="page-content">
  <h1>JMp SmaRT COOP</h1>

  <div class="card">
    <h2>Χειροκίνητος Έλεγχος</h2>
    <div class="flex-row">
```

```
    <button class="danger-button" onclick="manualFeed()">Feed</button>
    <button class="danger-button" onclick="toggleDoor()">Door</button>
  </div>
</div>

<div class="card" id="output">Φόρτωση...</div>

<div class="card">
  <h2>Ρυθμίσεις</h2>

  <div class="flex-row">
    <label>Weather Protection:
      <label class="switch">
        <input type="checkbox" id="weatherToggle">
        <span class="slider"></span>
      </label>
    </label>

    <label>Auto Door:
      <label class="switch">
        <input type="checkbox" id="autoDoorToggle">
        <span class="slider"></span>
      </label>
    </label>

    <label>Feeder:
      <label class="switch">
        <input type="checkbox" id="feederToggle">
        <span class="slider"></span>
      </label>
    </label>
  </div>
```

```
    <h3>Ώρα Ταιΐσματος</h3>
    <div class="flex-row">
      <input type="number" id="feederHour" min="0" max="23"> :
      <input type="number" id="feederMin" min="0" max="59">
    </div>

    <h3>Ώρα Άνοιγμα Πόρτας</h3>
    <div class="flex-row">
      <input type="number" id="openHour" min="0" max="23"> :
      <input type="number" id="openMin" min="0" max="59">
    </div>

    <h3>Ώρα Κλείσιμο Πόρτας</h3>
    <div class="flex-row">
      <input type="number" id="closeHour" min="0" max="23"> :
      <input type="number" id="closeMin" min="0" max="59">
    </div>

    <div style="text-align: center;">
    <button onclick="sendSettings()">Αποθήκευση Ρυθμίσεων</button>
  </div>
</div>
</div>

<div id="countdown-timer"></div>

<script type="module">
  import { initializeApp } from "https://www.gstatic.com/firebasejs/10.12.0/firebase-app.js";
  import { getDatabase, ref, onValue, set } from "https://www.gstatic.com/firebasejs/10.12.0/firebase-database.js";

const firebaseConfig = {
```

```
  apiKey: "AIzaSyDUMuRZCncdTnVhhVRHwDd_kjY2vJW6ST0",
  authDomain: "jmp-coop.firebaseapp.com",
  databaseURL: "https://jmp-coop-default-rtdb.europe-west1.firebasedatabase.app",
  projectId: "jmp-coop",
  storageBucket: "jmp-coop.firebasestorage.app",
  messagingSenderId: "390471597399",
  appId: "1:390471597399:web:584eeb9a2dffe0c9c81212"
};
  const app = initializeApp(firebaseConfig);
  const db = getDatabase(app);

  const dataRef = ref(db, "/esp32/data");
  const cmdRef = ref(db, "/esp32/commands");

  onValue(dataRef, (snap) => {
    const d = snap.val();
    console.log("Firebase data:", d); // Log the entire data object
    console.log("Door status value:", d["Door Position"]); // Log the door position value
    let warnings = "";

    if (d.WARNINGS == 1) {
      warnings = `<div class="warning">⚠️ ΠΡΟΕΙΔΟΠΟΗΣΗ!</div>`;
    }

    let chickenStatus = '';
    if (d["Chiken all inside"] === 1) {
        chickenStatus = '<p style="text-align: center;">όλες οι κοτες είναι μέσα</p>';
    } else if (d["Chiken all Outside"] === 1) {
        chickenStatus = '<p style="text-align: center;">όλες οι κοτες είναι έξω</p>';
    } else {
        chickenStatus = `<span style="text-align: center;">Chicken Inside: ${d["Chiken
Inside"]}</span> <span style="text-align: center;">Chicken Outside: ${d["Chiken
Outside"]}</span>`;
```

```
  }

  let doorStatusHtml = '';
  const doorStatus = d["Door Position"];
  if (doorStatus !== undefined && doorStatus !== null) { // Added check for undefined and null
    if (doorStatus == 1) { // Changed to loose equality
        doorStatusHtml = '<p style="text-align: center;">Door: Open</p>';
    } else if (doorStatus == 0) { // Changed to loose equality
        doorStatusHtml = '<p style="text-align: center;">Door: Closed</p>';
    }
  }

  document.getElementById("output").innerHTML = `
    <p style="text-align: center;"><strong>Ώρα:</strong> ${d.hour}:${d.minute}</p>
    <div class="chicken-status-box">${chickenStatus}${doorStatusHtml}</div>
    ${warnings}
  `;

  document.getElementById("weatherToggle").checked = d["WEATHER PROTECTION"];
  document.getElementById("autoDoorToggle").checked = d["AUTO DOOR"];
  document.getElementById("feederToggle").checked = d["FEEDER"];

  document.getElementById("feederHour").value = d.setHourF;
  document.getElementById("feederMin").value = d.setMinF;
  document.getElementById("openHour").value = d.setHourOD;
  document.getElementById("openMin").value = d.setMinOD;
  document.getElementById("closeHour").value = d.setHourCD;
  document.getElementById("closeMin").value = d.setMinCD;
});

window.sendSettings = async () => {
  const data = {
```

```
      type: "update_settings",
      weather: document.getElementById("weatherToggle").checked,
      autoDoor: document.getElementById("autoDoorToggle").checked,
      feeder: document.getElementById("feederToggle").checked,
      feederHour: parseInt(document.getElementById("feederHour").value),
      feederMin: parseInt(document.getElementById("feederMin").value),
      openHour: parseInt(document.getElementById("openHour").value),
      openMin: parseInt(document.getElementById("openMin").value),
      closeHour: parseInt(document.getElementById("closeHour").value),
      closeMin: parseInt(document.getElementById("closeMin").value),
      timestamp: Date.now()
    };
    await set(cmdRef, data);
    alert("Οι ρυθμίσεις εστάλησαν!");
  };

  window.manualFeed = async () => {
    await set(cmdRef, {
      type: "manual_feed",
      timestamp: Date.now()
    });
    alert("Εστάλη εντολή για τάισμα.");
  };

  window.toggleDoor = async () => {
    await set(cmdRef, {
      type: "toggle_door",
      timestamp: Date.now()
    });
    alert("Εστάλη εντολή για πόρτα.");
  };
</script>
```

```
<script>
    const correctPassword = '511069';
    const loginDuration = 30 * 60 * 1000; // 30 minutes in milliseconds
    let countdownInterval = null;

    function showPageContent() {
        document.getElementById('password-protection').style.display = 'none';
        document.getElementById('page-content').style.display = 'block';
        document.getElementById('countdown-timer').style.display = 'block'; // Show timer
        startCountdown();
    }

    function showPasswordProtection() {
        document.getElementById('password-protection').style.display = 'block';
        document.getElementById('page-content').style.display = 'none';
        document.getElementById('password-input').value = ''; // Clear input
        document.getElementById('error-message').textContent = ''; // Clear error message
        document.getElementById('countdown-timer').style.display = 'none'; // Hide timer
        if (countdownInterval) {
            clearInterval(countdownInterval);
            countdownInterval = null;
        }
    }

    function checkPassword() {
        const password = document.getElementById('password-input').value;
        const errorMessage = document.getElementById('error-message');

        if (password === correctPassword) {
            localStorage.setItem('loggedInTimestamp', Date.now());
            showPageContent();
```

```
        } else {
            errorMessage.textContent = 'Λάθος κωδικός. Προσπαθήστε ξανά.';
        }
    }

    function updateCountdown() {
        const loggedInTimestamp = localStorage.getItem('loggedInTimestamp');
        if (!loggedInTimestamp) {
            showPasswordProtection();
            return;
        }

        const currentTime = Date.now();
        const timeElapsed = currentTime - parseInt(loggedInTimestamp);
        const timeLeft = loginDuration - timeElapsed;
        const countdownTimer = document.getElementById('countdown-timer');

        if (timeLeft <= 0) {
            localStorage.removeItem('loggedInTimestamp');
            showPasswordProtection();
        } else {
            const minutes = Math.floor(timeLeft / 60000);
            const seconds = Math.floor((timeLeft % 60000) / 1000);
            countdownTimer.textContent = `\u039b\u03AE\u03BE\u03B7: ${minutes}m ${seconds}s`; // Corrected character
        }
    }

    // Check login status on page load
    window.onload = function() {
        const loggedInTimestamp = localStorage.getItem('loggedInTimestamp');

        if (loggedInTimestamp) {
```

```
            const currentTime = Date.now();
            if (currentTime - parseInt(loggedInTimestamp) < loginDuration) {
                showPageContent();
            }

        } else {
            // Timestamp expired
            localStorage.removeItem('loggedInTimestamp');
            showPasswordProtection();
        }
    };

</script>

</body>
</html>
```

ΠΑΡΑΡΤΗΜΑ Γ : BOM (BILL OF MATERIALS)

ESP32C6-DEVKIT-N8	9.90 €
RTC DS1307	1.90 €
I2C MODULE	1.20 €
16X2 LCD	3.20 €
2X RDM6300	13.40 €
RAIN SESNOR	2.30 €
2X TT DC MOTOR	6.40 €
L9110S MOTOR DRIVER	2.50 €
IR BREAK BEAM SENSOR	2.40 €
2X REED SWITCH	1.40 €
3X PUSH BUTTONS	0.40 €
5X RESISTOR	0.10 €
3X CAPACITOR	0.10 €
1X LED	0.08 €
1X NTC THERMISTOR	0.80 €
1X LDR RESISTOR	0.20 €
1X 3V RELAY	2.10 €
1X LIQUID LEVEL SENSOR	3.00 €
1X 12V-30W HEATER	8.00 €
PCB BOARD	3.20 €
5X RFID 125KHZ TAGS	4.00 €
3D PRINTS (ASA)	15.00 €
ΝΤΙΖΑ-ΑΞΟΝΕΣ-ΠΑΞΙΜΑΔΙΑ	10.00 €
ΓΩΝΙΕΣ-ΚΟΙΛΟΔΟΚΟΙ-ΠΛΕΓΜΑ	80.00 €
ΦΩΤΟΒΟΛΤΑΙΚΟ 30W	35.00 €
ΡΥΘΜΙΣΤΗΣ ΦΟΡΤΙΣΗΣ	15.00 €
ΜΠΑΤΑΡΙΑ 12V	30.00 €
ΣΥΝΟΛΙΚΟ ΚΟΣΤΟΣ :	251.58 €